H. J. Kottmann

DIE AUTOMATISIERTE BELEGUNGSPLANUNG VON GROSSRECHENANLAGEN MIT HILFE OPTIMIERENDER STRATEGIEN UNTER BERÜCKSICHTIGUNG VON DATENABHÄNGIGKEITEN UND ABGABETERMINEN

1973

DIE AUTOMATISIERTE BELEGUNGSPLANUNG VON GROSSRECHENANLAGEN MIT HILFE OPTIMIERENDER STRATEGIEN UNTER BERÜCKSICHTIGUNG VON DATENABHÄNGIGKEITEN UND ABGABETERMINEN

Inaugural - Dissertation
zur Erlangung des akademischen Grades
eines Doktors der Wirtschaftswissenschaft
durch den Fachbereich
Wirtschafts- und Sozialwissenschaften
der Westfälischen Wilhelms-Universität
zu Münster

vorgelegt von
Dipl.-Volkswirt Hans-Jürgen Kottmann
aus Weidenau/Sieg
1973

UNIV.
LEIDEN
BIBL.

Erster Berichterstatter	:	Prof. Dr. Bleymüller
Zweiter Berichterstatter	:	Prof. Dr. Wagner
Dekan	:	Prof. Tilly Ph.D.
Tag der mündlichen Prüfung	:	6. Februar 1973

INHALTSVERZEICHNIS

		Seite
VERZEICHNIS DER TABELLEN		VI
VERZEICHNIS DER ABBILDUNGEN		VIII
VERZEICHNIS DES ANHANGS		XI
1	EINFÜHRUNG	1
2	DIE MULTIPROGRAMMING-BETRIEBSSYSTEME EINIGER HERSTELLER	11
2.1	Allgemeine Einführung	11
2.2	Die Funktionen eines Betriebssystems	15
2.3	IBM: Operating System (OS)	18
2.4	UNIVAC: Executive System (EXEC 8)	34
2.5	SIEMENS: Platte-Betriebssystem (PBS)	41
3	AUFBAU EINER JOBSTAMMDATEI	49
3.1	Die Aufgaben einer Jobstammdatei	49
3.2	Die Definition eines Jobstammsatzes	52
3.3	Auswertungsmöglichkeiten	64
4	DURCHFÜHRUNG DER JOB-ABLAUFPLANUNG FÜR EINE PLANUNGSPERIODE OHNE BERÜCKSICHTIGUNG VON KAPAZITÄTEN	71
4.1	Die Darstellung von Jobstrukturen mit Hilfe von Netzwerken	71
4.2	Rangrechnung und Blockbildung	82

Seite

4.3 Die Terminplanung für den Jobablaufplan einer Planungsperiode 94

4.4 Beschreibung eines Programms zur Job-Ablaufplanung 100

5 DARSTELLUNG DER KAPAZITÄTSAUSGLEICHSVERFAHREN 105

5.1 Definition der Zielvorstellung für die Belegungsplanung und Diskussion verschiedener Kapazitätsplanungskriterien 105

5.2 Exakte Verfahren 117

5.3 Heuristische Verfahren 130

5.4 Einige Erfahrungen mit Prioritätsregelverfahren 146

6 BESCHREIBUNG EINES HEURISTISCHEN VERFAHRENS FÜR DIE OPTIMIERENDE BELEGUNGSPLANUNG EINER IBM/370-165 IN EINEM TECHNISCH/KAUFMÄNNISCH ORIENTIERTEN RECHENZENTRUM 155

6.1 Überlegungen zum organisatorischen Ablauf einer automatisierten Tages- und Belegungsplanung 155

6.2 Erläuterung der Anlagenkonfiguration und der bei der Belegungsplanung berücksichtigten Systemelemente 160

6.3 Beschreibung des Belegungsplanungsverfahrens .. 166

6.4 Die Effizienz des Planungsverfahrens 193

7 DISKUSSION EINIGER IM ZUSAMMENHANG MIT DER EINFÜHRUNG DES ABLAUF- UND BELEGUNGSPLANUNGSVERFAHRENS AUFTRETENDER PROBLEME 201

8 SCHLUSSBETRACHTUNG 211

	Seite
LITERATURVERZEICHNIS	215
ANHANG	219
VERZEICHNIS DER ABKÜRZUNGEN	229
VERZEICHNIS DER ABKÜRZUNGEN ZU DEN AB-LAUFDIAGRAMMEN IN KAPITEL 6	231

VERZEICHNIS DER TABELLEN

		Seite
Tab. 1	Entwicklung der Stückzahlen von Großrechenanlagen in der BRD	4
Tab. 2	Marktanteile der Hersteller von Großrechenanlagen in der BRD	4
Tab. 3	Entwicklung der Anzahl und des Anteils der im Multiprogramming betriebenen Anlagen von 1970 bis 1978	7
Tab. 4	Entwicklung der Schaltzeiten je Rechenoperation	12
Tab. 5	Übersicht über die Angaben in der Parameterkarte für das Programm DAB 3	46
Tab. 6	Verteilung der Verarbeitungszeiten von 1707 Routinejobs eines Großrechenzentrums	67
Tab. 7	Verteilung der Druckausgabeklassen von 1707 Jobs	68
Tab. 8	Verteilung der maximal in einem Job belegten Magnetbandeinheiten	69
Tab. 9	Beispiel der Darstellung eines Graphen in Listenform	73
Tab. 10	Beispiel der Matrixdarstellung eines Graphen	75
Tab. 11	File-Katalogeintragung für eine Magnetbanddatei	78
Tab. 12	Automatische Bestimmung von Vor- und Folgejobs	79
Tab. 13	Beispiel zur Rangrechnung	83
Tab. 14	Matrix zu Rangrechnungs-Verfahren 1	84
Tab. 15	Ergebnis der Rangrechnung für das Beispiel in Tab. 13	85
Tab. 16	Tabelle zur Rangrechnung mit Hilfe des 2. Verfahrens	87

		Seite
Tab. 17	Beispiel zur Bestimmung der Zahl der Durchläufe .	88
Tab. 18	Beispiel zur Zyklusbildung	89
Tab. 19	Feststellung der Abhängigkeiten für das Beispiel aus Tab. 18	90
Tab. 20	Feststellung der Schleife für das Beispiel aus Tab. 18	92
Tab. 21	Ergebnisse der Zeitrechnung aus Abb. 19 in Minuten	98
Tab. 22	Terminplan für den Jobblock aus Abb. 19	99
Tab. 23	Verteilung von 45 Jobblöcken nach der Anzahl der Jobs pro Jobblock	106
Tab. 24	Zeitrechnung für die Jobstruktur aus Abb. 28	135
Tab. 25	Entscheidungssituationen beim Verfahren der Variationen-Enumeration für das Beispiel aus Abb. 28	136
Tab. 26	Zeitrechnung und Bestimmung der Restzeiten für Abb. 38	183
Tab. 27	Darstellung der möglichen Ausgangsbedingungen und der erforderlichen Maßnahmen in der dritten Stufe des Planungsverfahrens nach der Auswahl von j_t Jobs mit der kleinsten Bandeinheitenbelegung und der größten Restzeit (j_t = Anzahl j der zum Zeitpunkt t zusätzlich einzuplanenden Jobs .	186
Tab. 28	Einplanungsunterlage eines Konsoloperators in einem Rechenzentrum ohne Belegungsplanung für das Beispiel aus Abb. 32	196

VERZEICHNIS DER ABBILDUNGEN

		Seite
Abb. 1	Ausnutzung der Zentraleinheit bei einer Verarbeitung im Uniprogramming	14
Abb. 2	Ausnutzung der Zentraleinheit bei einer Verarbeitung im Multiprogramming	14
Abb. 3	Aufbau des Operating Systems (OS).....	20
Abb. 4	Elemente des Job Management	23
Abb. 5	Problematik des core fregmentation	30
Abb. 6	Ein prioritätsgesteuertes Planungssystem	31
Abb. 7	Beispiel eines Jobstammblattes für Job J 999999..	54
Abb. 8	Aufbau des Terminsteuerworts	56
Abb. 9	Verarbeitungszeit-Übersicht für eine Planungsperiode	66
Abb. 10	Beispiel eines einfachen Graphen	73
Abb. 11	Beispiel eines vollständigen Graphen	74
Abb. 12	Beispiel eines Baums	74
Abb. 13	Darstellung der in Tab. 12 dargestellten Jobabhängigkeiten als Netzwerk	80
Abb. 14	Netzwerk des Beispiels aus Tab. 13 mit Rangeinteilung	86
Abb. 15	Beispiel zur Veranschaulichung der maximal erreichbaren Rangziffer bei 7 Jobs (A 1, ..., A 7) und den Rängen (R 0, ..., R 6)	88
Abb. 16	Graph der Tab. 17	88
Abb. 17	Graph der Tab. 19	90

		Seite
Abb. 18	Beispiel zur Job-Blockbildung für 18 Jobs 1 - 18	93
Abb. 19	Beispiel zur Zeitrechnung	95
Abb. 20	Programmstruktur des Ablaufplanungsprogramms	101
Abb. 21	Jobstruktur des in Anhang 2 gezeigten Jobblocks	104
Abb. 22	Verfügbare Kapazität von acht Bandeinheiten für einen Zeitraum von acht Stunden	110
Abb. 23	Beispiel einer Bandeinheitenbelegung für 4 Jobs	112
Abb. 24	Methoden zur Lösung von Reihenfolgeproblemen	117
Abb. 25	Beispiel einer unterbrochenen Jobausführung	121
Abb. 26	Beispiel zur Darstellung von Jobabhängigkeiten in einem Modell der ganzzahligen, linearen Optimierung	122
Abb. 27	Beispiel zur Bestimmung der Restbelegungsdauer von Bandeinheiten für noch nicht eingeplante Jobs im Rahmen der begrenzten Enumeration	129
Abb. 28	Jobstruktur für die Durchführung der Variationen-Enumeration	134
Abb. 29	Einteilung der mehrstufigen Prioritätsregelverfahren	138
Abb. 30	Ablaufdiagramm eines mehrstufigen, statisch-seriellen Einplanungsverfahrens mit Warteliste	141
Abb. 31	Ablaufdiagramm des mehrstufigen, dynamisch-parallelen Einplanungsverfahrens	143
Abb. 32	Test-Beispiel für die Durchführung der Belegungsplanung mit Hilfe verschiedener Prioritätsregelkombinationen	150
Abb. 33	Ergebnis der Einplanung für das Beispiel aus Abb. 32 bei der Priotitätsregelkombination 1. hohe Blockdauer, 2. niedrige gesamte Pufferzeit, 3. hohe Verweilzeit	152

		Seite
Abb. 34	Planungssystem für die Durchführung der Stapelverarbeitung	156
Abb. 35	Organisationsstruktur eines Rechenzentrums	159
Abb. 36	Konfiguration des Systemverbundes bestehend aus einer Anlage des Typs IBM/370-165 und einer Anlage des Typs /370-155	161
Abb. 37	Ablaufdiagramm für die automatische Festlegung des Engpaßelements und die Auswahl des für die Belegungsplanung günstigsten Zweiges	179
Abb. 38	Jobblock zur Bestimmung der Restzeiten	183
Abb. 39	Beispiel zur Erläuterung des Unterschiedes zwischen dem Nachlauf und der Restzeit eines Jobs	184
Abb. 40	Ablaufdiagramm zur Auswahl eines zu streichenden Jobs unter der Voraussetzung, daß die Bandeinheitenbelegung größer ist als die Verfügbarkeit	188
Abb. 41	Ablaufdiagramm zur Verbesserung der Bandeinheitenbelegung	192
Abb. 42	Integriertes Ablaufdiagramm des Planungsprogramms	194
Abb. 43	Voraussichtliches manuelles Einplanungsergebnis für die Bandeinheiten auf Grund der Einplanungsunterlage in Tab. 28 durch einen Operator	198
Abb. 44	Belegungsplan für das Beispiel in Abb. 32 auf Grund des vom Verfasser entwickelten Planungsverfahrens bei drei Regions und sieben Bandeinheiten	200

VERZEICHNIS DES ANHANGS

		Seite
Anhang 1	Beispiel eines Ausdrucks der Monatsplanungsübersicht für einen bestimmten Tag	220
Anhang 2	Beispiel einer Listenausgabe für die Job-Ablaufplanung	221
Anhang 3	Ergebnis der Zeitrechnung in Minuten für das Beispiel in Abb. 32	222
Anhang 4	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedrige Pufferzeit, 2. niedrige Verweilzeit, und 3. niedriger spätester Endzeitpunkt	223
Anhang 5	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedrige Pufferzeit, 2. hohe Verweilzeit und 3. niedriger frühester Anfangszeitpunkt	224
Anhang 6	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. hohe Blockdauer, 2. hohe Verweilzeit und 3. niedrige Pufferzeit	225
Anhang 7	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. hohe Verweilzeit, 2. niedrige Pufferzeit und 3. niedriger spätester Endzeitpunkt	226
Anhang 8	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedriger frühester Anfangszeitpunkt, 2. niedrige Pufferzeit und 3. hohe Verweilzeit	227
Anhang 9	Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedriger frühester Endzeitpunkt, 2. niedrige Pufferzeit und 3. hohe Verweilzeit	228

1 EINFÜHRUNG

Wohl kaum ein anderer Wirtschaftszweig hat in den letzten 25 Jahren eine so stürmische Entwicklung zu verzeichnen gehabt wie der Bereich der elektronischen Datenverarbeitung (EDV). Bisher sind drei sog. Generationen von Datenverarbeitungsanlagen zum Einsatz gekommen: 1946 die erste Generation, bei der noch Relaisschaltungen und Röhren verwendet wurden, 1958 die zweite, bei der man Transistoren benutzte, und 1966 die dritte, bei der man auf miniaturisierte Schaltkreise überging. Zu Beginn des Jahres 1969 waren weltweit mehr als 80.000 automatische Datenverarbeitungsanlagen (DVA) eingesetzt, davon etwa 55.000 in den Vereinigten Staaten und fast 18.000 in westeuropäischen Ländern. Die Zuwachsraten für die Anzahl der eingesetzten Rechner liegen dabei in den USA bei zur Zeit etwa 20 %, in einigen Ländern Westeuropas auf Grund eines starken Nachholbedarfs sogar bei bis zu 50 % jährlich[1].

Der Einsatz der elektronischen Rechenanlagen erweist sich dabei immer mehr als einer der maßgebenden Faktoren der zukünftigen wirtschaftlichen Entwicklung. Die Anwendungsgebiete der EDV erstrecken sich heute auf nahezu alle Bereiche eines Unternehmens. Neben dem Einsatz in fast "klassischen" Bereichen Buchhaltung, Gehalts- und Lohnabrechnung sowie der Materialwirtschaft und Lagerhaltung wird die EDV in naher Zukunft verstärkt auch auf dem Gebiet der Produktionsplanung und -steuerung, der Investitionsrechnung, dem Budgetwesen und der Unternehmensplanung eingesetzt werden.

---

1) DIEBOLD DEUTSCHLAND GMBH, Möglichkeiten und Grenzen der elektronischen Datenverarbeitung, Voruntersuchung, Teil 1, Frankfurt/Main, März 1970, S. 1; im folgenden zitiert als DIEBOLD STUDIE, Teil 1 - 3.

Der Umfang der Arbeiten, der damit vor allem in Großunternehmen auf die Datenverarbeitung zukommt, erfordert immer größere und schnellere EDV-Anlagen und verlangt vor allem von den in der Datenverarbeitung Beschäftigten die ständige Bereitschaft, sich mit neuen Verarbeitungsformen, wie z.B. Multiprogramming, Real-Time-Verarbeitung und Time Sharing, vertraut zu machen und umzulernen.

Die Entwicklung auf dem Computermarkt der Bundesrepublik Deutschland wird voraussichtlich in zwei Richtungen verlaufen. Auf der einen Seite wird die elektronische Datenverarbeitung in mittlere und kleinere Unternehmen vordringen. Es werden damit etwa 40.000 Unternehmen mit je 50 - 500 Beschäftigten angesprochen, die bisher nur in sehr geringem Umfang von der Automatisierungswelle durch die Datenverarbeitung erfaßt wurden, da der Aufwand für die Installation eines Rechners bisher für sie zu hoch war. Auf der anderen Seite wird sich die Zahl der eingesetzten Anlagen bei den bisherigen Anwendern erheblich erhöhen. Dabei werden vermutlich viele kleinere Anlagen dezentralisiert aufgestellt werden, die einmal der Datenerfassung, zum anderen aber auch der Durchführung bestimmter Routinearbeiten einfacherer Art dienen und die gleichzeitig als Eingabestationen für Großrechnersysteme fungieren, auf denen komplexe Auswertungen und Planungsrechnungen durchgeführt werden.

Man rechnet damit, daß 1978 in der BRD ca. 25.000 Computer aufgestellt sein werden im Vergleich zu etwa 8.000 Computern Anfang 1971[1)].

Neben dieser rein quantitativen Prognose für die Zahl der insgesamt installierten Rechner ist auch die strukturelle Entwicklung für einzelne Größenklassen von Interesse.

Die Untersuchung der DIEBOLD DEUTSCHLAND GMBH zeigt, daß Anlagen mit einem monatlichen Mietwert von bis zu DM 8.000,-- (Klasse I)

---

1) DIEBOLD STUDIE 1, a.a.O., S. 55, 56.

und Anlagen mit einem monatlichen Mietwert von DM 8.001,-- bis DM 20.000,-- (Klasse II) zahlenmäßig am stärksten zunehmen werden. "Ansätze dazu zeigen sich schon in der Größenkonstellation der letzten Jahre: zum Jahresanfang 1969 sind 21 % des stückmäßigen Bestandes der Klasse I und 43 % der Klasse II zuzurechnen. In den nächsten Jahren werden die Anlagen dieser beiden Größenklassen annähernd 70 % einnehmen, während zum Jahresanfang 1966 ihr Anteil noch bei 50 % lag."[1)]

Die Expansion in der Klasse I wird dabei neben der Neuanschaffung von Computern in Klein- und Mittelbetrieben vor allem auf Prozeßrechner und an Großcomputer angeschlossene Satellitenrechner zurückzuführen sein.

Datenverarbeitungsanlagen mit einem monatlichen Mietwert von DM 20.001,-- bis DM 80.000,-- (Klassen III u. IV) werden in den nächsten Jahren anteilsmäßig und ab 1975 auch absolut zurückgehen, weil vor allem die kleineren Anlagen bis DM 20.000,-- deutliche Leistungsverbesserungen aufweisen werden.

Großanlagen mit einem monatlichen Mietwert von über DM 80.000,-- (Klasse V) werden absolut zunehmen und dabei ihren Anteil, der etwa bei 2 % liegt, halten können.

Da in dieser Arbeit in der Hauptsache Probleme behandelt werden, die beim Einsatz von Großanlagen auftreten, soll die Zeitreihe für die bisherige und die voraussichtliche Entwicklung der Stückzahlen dieser Klasse in der BRD gesondert aufgeführt werden:

---

1) DIEBOLD STUDIE 1, a.a.O., S. 58.

Tab. 1 : Entwicklung der Stückzahlen von Großrechenanlagen in der BRD

Jahresanfang	1959	60	61	62	63	64	65	66	67	68	69	70	71
Anzahl	1	2	12	19	32	44	40	45	44	70	101	130	160

72	73	74	75	76	77	78
200	240	290	340	400	450	500

1)

(ab 1970 Schätzwerte)

Die Marktanteile der einzelnen Hersteller ergaben für die Großanlagen (Größenklasse V) im Jahr 1969 das in Tab. 2 dargestellte Bild.

Tab. 2 : Marktanteile der Hersteller von Großrechenanlagen in der BRD

Hersteller	Marktanteil
1. IBM	66 %
2. Univac	17 %
3. CDC	10 %
4. Siemens	3 %
5. AEG-Telefunken	3 %
6. ICL	1 %

Die IBM hat auf dem Markt für Großrechensysteme demnach eine beherrschende Stellung inne, die voraussichtlich durch die Rechner der Serie /370 noch weiter gefestigt werden kann.

In einer früheren Studie der DIEBOLD-Gruppe[2], die vor allem auf Befragungen von Anwendern der elektronischen Datenverarbeitung basiert und in der etwa 40 % der installierten Großanlagen repräsentiert sind, wird ein Überblick über die Einsatzbereiche der automatischen Datenverarbeitung (Stand Mai 1968) gegeben. Danach entfielen 81 % der gesamten Verarbeitungszeit auf Routinearbeiten, 16 % auf Planungsarbeiten und 3 % auf die

1) DIEBOLD STUDIE 1, a.a.O., S. 24.

2) DIEBOLD DEUTSCHLAND GMBH, Einsatz und Stand der Datenverarbeitung in der Bundesrepublik Deutschland, Frankfurt/M., 1968, S. 8.

Verarbeitung von Unternehmensmodellen. Bezieht man auch die geplanten Vorhaben mit ein, verschieben sich die Relationen zugunsten der Planungs- und Modellarbeiten, nämlich: 71 % Routine, 24 % Planung, 5 % Modellverarbeitung. Die Routinearbeiten verteilen sich zu 70 % auf kaufmännische Arbeiten, zu 16 % auf technisch wissenschaftliche Arbeiten, zu 10 % auf Produktionssteuerung und zu 4 % auf Dokumentation.

Ein letzter Aspekt, der in diesem Zusammenhang noch einer genaueren Analyse bedarf, ist der momentane und der geplante Einsatz der Multiprogrammierung (Multiprogramming), deren Einsatz Voraussetzung für die in dieser Arbeit angestellten Überlegungen ist. "Die Multiprogrammierung stellt eine Betriebsweise des Computers dar, bei der mehrere voneinander unabhängige Programme parallel auf einer Rechenanlage bearbeitet werden, indem der gleichzeitige Ablauf durch ein besonderes Steuerprogramm ermöglicht wird. Dieses Verfahren erlaubt eine wirtschaftliche Anlagenausnutzung, vor allem, wenn rechenintensive Programme mit Arbeiten gekoppelt werden, die intensiven Datenverkehr zu peripheren Einheiten erfordern."[1] (Eine ausführlichere Erklärung dieser Verarbeitungsform soll Kapitel 2 vorbehalten bleiben.)

Für den bis etwa 1970 nur recht zögernd erfolgten Einsatz des Multiprogramming auf dazu geeigneten Anlagen gibt es mehrere Gründe. So waren bei einigen Benutzern ohne Zweifel die personellen und organisatorischen Voraussetzungen, die die Einführung dieser Verarbeitungsform erfordert, noch nicht gegeben. Das Hauptproblem bestand jedoch in der Bereitstellung benutzerfreundlicher und die Möglichkeiten des Multiprogramming voll unterstützender Betriebssysteme auf der Seite der Hersteller.

Bei dem bis vor drei bis vier Jahren ausschließlich angewandten Uniprogramming konnte immer nur ein Programm im Rechner abgewickelt werden. Heute können je nach Programmgröße und zur Verfügung stehendem Kernspeicher sowie der Anzahl der peripheren Geräte durchschnittlich 3-7,

---

1) DIEBOLD STUDIE 3, a.a.O., S. 18.

maximal bis zu 15 Programme zeitlich verzahnt im Rechner erledigt werden.

Für die Verwirklichung der Multiprogrammierung in einem Rechenzentrum sind drei verschiedene Methoden denkbar:

1. Laufende Einplanung der Jobs durch den Operator
   Diese Methode wird in den meisten Rechenzentren angewendet. Sie ist jedoch abzulehnen, da der Operator weder die Kernspeicher- und Peripherieanforderungen der Jobs kennen noch die häufig täglich wechselnden Abhängigkeiten der Jobs untereinander so beachten kann, daß die Anlage optimal genutzt wird.

2. Manuelle oder maschinelle Vorausplanung des Multiprogrammbetriebs
   Das Hauptargument gegen diese Art der Planung ist die beim Multiprogramming nur sehr schwer abschätzbare Laufzeit der Jobs, die dazu führen kann, daß solche Pläne nur kurzfristig aussagefähig sind. Dieser Einwand ist zwar schwerwiegend; es soll aber der Nachweis erbracht werden, daß die maschinelle Vorausplanung nach Meinung des Verfassers die mit dem geringsten Aufwand zu verwirklichende Möglichkeit bietet, zu befriedigenden Ergebnissen zu kommen.

3. Dynamische Anlagenbelegung bei maschinell gesteuertem Mehrprogrammbetrieb
   Dazu muß von seiten eines sich ständig in der Anlage befindlichen Programms, das auf Grund von Parameterkarten die Kapazitätsanforderungen aller Programme und die verfügbare Kapazität der Anlage kennt, in kurzen Zeitabständen überprüft werden, ob noch ein weiteres Programm geladen werden kann. Wird ein solches gefunden, so wird über die Konsolschreibmaschine ein entsprechender Ladeaufruf geschrieben.

Unabhängig von der angewendeten Methode erfordert die Multiprogrammierung für das Datenverarbeitungspersonal jedoch sehr genaue und vor allem schnell verfügbare Informationen über die zu verarbeitenden Programme, die

heute in den wenigsten Großrechenzentren vorliegen. Bisher hat sich nach Wissen des Verfassers nur die Firma SIEMENS mit der Lösung dieses dringenden Informationsproblems beschäftigt[1].

Aus dem Material der DIEBOLD-Untersuchung läßt sich ablesen, daß zu Beginn des Jahres 1968 etwa 200 EDV-Anlagen im Multiprogramming betrieben wurden. Der weitaus größte Teil der Benutzer ist dabei im Bereich der Industrie (90), des Handels (35) und der Banken (15) zu finden[2]. Der Schwerpunkt für den Einsatz des Multiprogramming liegt vor allem in größeren Unternehmen mit mehr als 1.000 Beschäftigten, denn von den 90 Industriebetrieben entfallen nur 5 auf Betriebe mit je 500 - 999 Beschäftigten. Von den 200 Anlagen entfallen auf die IBM etwa 40 %, auf UNIVAC 35 % und auf alle anderen Hersteller etwa 25 %. Diesen Prozentsätzen entsprechend werden in Kapitel 2 die Betriebssysteme von IBM und UNIVAC ausführlich erläutert.

Der Anteil der mit Multiprogrammierung betriebenen Anlagen an der Gesamtzahl wird von DIEBOLD bis 1978 wie folgt gesehen:[3]

Tab. 3 : Entwicklung der Anzahl und des Anteils der im Multiprogramming betriebenen Anlagen von 1970 bis 1978

Jahresanfang	Anteil der in Multiprogrammierung betriebenen Anlagen an der Gesamtzahl	Anzahl der in Multiprogrammierung betriebenen Anlagen an der Gesamtzahl
1970	10 %	650
1972	14 %	1.390
1975	18 %	3.080
1978	25 %	6.250

1) s. Kapitel 2.5

2) DIEBOLD STUDIE 3, a.a.O., S. 19.

3) DIEBOLD STUDIE 3, a.a.O., S. 21.

Das bedeutet, daß sich 1978 bei etwa 6.250 Anlagen das Multiprogramming-Verfahren durchgesezt haben wird. In der DIEBOLD-Untersuchung werden die Konsequenzen dieser Entwicklung wie folgt beschrieben:
"Die Untersuchung der wichtigsten Merkmale der Multiprogrammierung zeigt, daß neben dem allgemeinen Trend zu kleineren Anlagen mit einfacherer Bedienung und standardisierten Softwarepaketen auf der anderen Seite auch neuartige Betriebsverfahren entwickelt werden und Verbreitung finden, die zumindest in einer längeren Anlaufphase auf Grund ihrer Komplexität erhebliche Anforderungen an das Personal und die Organisation des Datenverarbeitungsbereichs stellen."[1)]
Es wird hier also klar zum Ausdruck gebracht, daß mit wachsendem Arbeitsumfang und dementsprechend größeren Rechnern die Lösung eines Problems immer dringender wird, mit dem man sich bisher so gut wie gar nicht auseinandergesetzt hat, nämlich mit der täglichen Einsatzplanung eines Computers, der im Multiprogramming-Betrieb gefahren wird.

Auch in der Fachpresse liest man in der letzten Zeit viel über die mangelnde Einsatzplanung der EDV. Schlagworte von "der Unzufriedenheit des Managements mit der EDVA", "der ineffizienten Datenverarbeitung" und "der teuersten Fehlinvestition" sind in diesem Zusammenhang häufig zu lesen[2)]. Die Berichte stellen dabei allerdings fast ausschließlich darauf ab, daß die EDV im Hinblick auf die mit ihrer Hilfe erledigten Problemstellungen nicht effizient genug eingesetzt wird, d.h. darauf, daß man die Möglichkeiten nicht konsequent ausnutzt, die moderne Großrechensysteme bieten (z.B. integrierte Produktions- und Unternehmensplanung).

Neben diesem Aspekt sollte aber noch ein anderer Ansatzpunkt Beachtung finden, der für den rationellen Einsatz der EDV von Bedeutung ist und bei dem es um die Anlage selbst geht. Ziel jedes Rechenzentrums sollte es sein, die Anschaltzeit des Systems zu minimieren, in dem es zur Erledigung der

---

1) DIEBOLD STUDIE 3, a.a.O., S. 22.

2) Vgl. z.B. WIRTSCHAFTSWOCHE Nr. 29, Jahrg. 1971, S. 55 ff und Nr. 51, 1971 S. 25 ff.

Aufgaben laufen muß. Die Anschaltzeit der Anlage ist von zwei Faktoren abhängig, nämlich erstens von der Effizienz der Programmierung, einer vor allem von der Qualität und dem Ausbildungsstand der Mitarbeiter abhängigen Größe, und zweitens von der Belegungsplanung der Datenverarbeitungsanlage selbst.

Wie wichtig die Lösung dieses Problems für die Datenverarbeitung ist, zeigte sehr deutlich eine von einem großen Computerhersteller 1970 durchgeführte Vortragsveranstaltung über die Führung von Großrechenzentren. Die Diskussion im Anschluß an den Vortrag drehte sich fast ausschließlich um die Frage der täglichen Belegungsplanung und Arbeitsvorbereitung für Großrechensysteme, ohne daß von der Herstellerseite eine befriedigende Lösung angeboten werden konnte.

Mit dem Begriff Belegungsplanung wird folgende Aufgabenstellung beschrieben: Auf einer DVA muß in einem vorgegebenen Zeitraum (z.B. an einem Tag) eine bestimmte, im voraus bekannte Anzahl von Aufgaben (sog. Jobs) erledigt werden. Zwischen diesen Aufgaben bestehen in der Regel Abhängigkeiten, die unter anderem auf die Fortschreibung gemeinsam benutzter Dateien zurückzuführen sind. Die Abhängigkeiten wechseln ständig und die Arbeitsvorbereitung steht täglich vor der Aufgabe, die auftretenden Reihenfolgeprobleme zu lösen. Es gilt Reihenfolgen zu finden, die die Abhängigkeiten einhalten. Bei etwa 150 - 200 Jobs pro Tag ist diese Aufgabe manuell in einem vertretbaren Zeitraum nicht mehr zu bewältigen. In den Kapiteln 3 und 4 wird - in Anlehnung an die Graphentheorie - ein Verfahren vorgeschlagen, das eine automatische Lösung des Reihenfolgeproblems gestattet.

Die einzelnen Jobs erfordern im Multiprogramming zu ihrer Erledigung jeweils einen Teil der DVA. Sie benötigen die Zentraleinheit, Kernspeicherplatz, Magnetband- und/oder Magnetplatteneinheiten, Kartenleser, Drucker usw. Für eine effiziente Nutzung der EDV kommt es darauf an, die einzelnen Jobs so zu kombinieren, daß vorgegebene Bereitstellungs-

und Abgabetermine eingehalten, die bestehenden Abhängigkeiten nicht verletzt und die peripheren Geräte sowie der Kernspeicher möglichst optimal genutzt werden.

Betrachtet man die erwartete technische Entwicklung auf dem Computermarkt, so dürfte sicher sein, daß sich die Leistung der Zentraleinheit in den nächsten Jahren auf das Acht- bis Zehnfache des heutigen Standes erhöhen wird[1]. Selbst wenn es gelingt, die Zugriffszeiten der externen Speicher bedeutend herabzusetzen, werden diese trotzdem in verstärktem Maß die kapazitiven Engpaßfaktoren bei der Datenverarbeitung sein. Die möglichst optimale Verteilung dieser Engpaßfaktoren auf die zu erledigenden Jobs stellt sich damit in naher Zukunft in jedem Großrechenzentrum. In den Kapiteln 5 und 6 wird deshalb ein vom Verfasser entwickeltes Verfahren vorgestellt, das dieses Zuordnungsproblem unter Beachtung bestimmter optimierender Strategien löst, wie sie u.a. aus der Kapazitätsausgleichsplanung in der Netzplantechnik bekannt sind. Dabei wird vor allem eine Lösung angestrebt, die sich in der Praxis durchführen läßt, da theoretisch exakte Optima bei der Größenordnung dieses Problems mit Hilfe der heute zur Verfügung stehenden DVA nicht möglich sind. Die Optimierung kann sich natürlich nur auf sog. "externe" Kapazitäten wie Magnetband- und Magnetplatteneinheiten beziehen und nicht auf die Zuordnung der Zentraleinheit oder der Input/Output-Kanäle, die immer nur Sekunden bzw. Sekundenbruchteile von einem Job belegt werden.

In Kapitel 7 werden die Probleme der zur Ablauf- und Belegungsplanung vorgeschlagenen Verfahren diskutiert.

In Kapitel 8 werden schließlich die über den Rahmen der täglichen Belegungsplanung hinausgehenden Möglichkeiten des Planungsverfahrens für die ausgeglichene Konfigurierung eines Rechnersystems angedeutet.

---

1) DIEBOLD STUDIE 3, a.a.O., S. 3.

2 DIE MULTIPROGRAMMING BETRIEBSSYSTEME EINIGER HERSTELLER

2.1 Allgemeine Einführung

In diesem Kapitel sollen die Betriebssysteme der Herstellerfirmen dargestellt werden, die bei den Großanlagen mit einem Mietwert von über DM 80.000,-- den größten Marktanteil besitzen[1]. Das sind also die Betriebssysteme von IBM, UNIVAC und CDC. Einen verhältnismäßig breiten Raum nimmt das Operating-System (OS) der IBM ein, da die Verträglichkeit dieses Betriebssystems mit dem vom Verfasser entwickelten Planungs- und Belegungsverfahren - allein schon vom Marktanteil der IBM her gesehen - für viele Benutzer von großem Interesse sein wird. Natürlich können die einzelnen Betriebssysteme nicht in ihrem gesamten Umfang beschrieben werden, sondern es muß eine Beschränkung auf die für die Zielrichtung der Arbeit wesentlichen Gesichtspunkte erfolgen. Auf die Darstellung des CDC-Betriebssystems SCOPE wird dabei zugunsten des SIEMENS-Betriebssystem "PBS" verzichtet, da SIEMENS kürzlich eine für diese Arbeit sehr interessante Neuerung in das Betriebssystem eingefügt hat.

Die Verarbeitungsform des "Multiprogramming", der "Multiprogrammierung" oder auch der "Multiprogrammverarbeitung" bedarf aber zunächst noch einer eingehenderen Erklärung, bevor die Funktionen eines Betriebssystems im einzelnen dargestellt werden.

Die Notwendigkeit zur Einführung des Multiprogramming wird deutlich, wenn man sich die Entwicklung der Schalt- und Zugriffszeiten ansieht, in denen eine Rechenoperation in der Zentraleinheit bzw. ein Lese- oder Schreibvorgang durchgeführt wird.

---

1) Vgl. S. 3 dieser Arbeit.

Tabelle 4: Entwicklung der Schaltzeiten je Rechenoperation[1)]

Computer-Generation	Bauelement	Schaltzeit je Rechenoperation
1	Relais (1938)	0,1 Sek.
1	Röhren (1946)	0,000 1 Sek.
2	Transistoren (1958)	0,000 001 Sek.
3	Monolithe (1966)	0,000 000 001 Sek.

Wie aus Tabelle 4 hervorgeht, ist die Verarbeitungsgeschwindigkeit der Zentraleinheiten von den Anfängen der EDV bis zu den Rechnern der 3. Generation auf das 100-Millionenfache gestiegen. Die Geschwindigkeit der peripheren Speicher (Magnetband-, Magnetplatten-, Magnettrommel-, Magnetkartenspeicher) hat sich im Vergleich hierzu dagegen nur in sehr bescheidenem Maße erhöhen lassen. So liegt die durchschnittliche Schaltzeit der Zentraleinheit einer IBM/370-165 bei ca. 80 Nanosekunden, die durchschnittliche Zugriffszeit eines IBM Magnetplattenspeichers 3330 dagegen bei ca. 30 Millisekunden. Die zukünftige Entwicklung läßt erwarten, daß die Schaltzeiten der Zentraleinheiten - in Abhängigkeit von den Abmessungen der Bausteine, aus denen sie bestehen - absolut in einem wesentlich stärkeren Umfang abnehmen werden als die Zugriffszeiten der herkömmlichen peripheren Speichermedien. Nur die Entwicklung neuartiger Speichermedien und Zugriffstechniken kann eine Angleichung der Schalt- und Zugriffszeiten ermöglichen.

Läuft nur ein Programm in einem Rechner ab, ergibt sich aus der dargestellten Relation der Zeiten, daß die Zentraleinheit häufig unbeschäftigt ist, weil sie auf die Beendigung eines Lese- oder Schreibvorganges warten muß. Je ein-/ausgabeintensiver ein Programm dabei ist, desto größer sind die Leerzeiten der Zentraleinheit und des Kernspeichers.

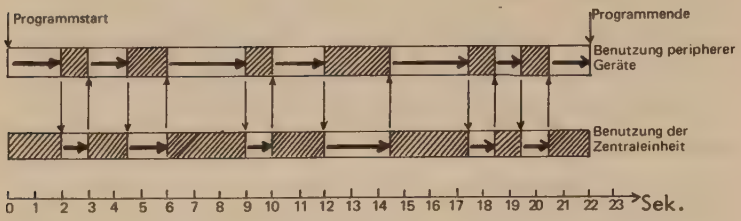
Bei einem Betriebssystem, mit dessen Hilfe eine Multiprogrammverarbeitung möglich ist, stehen ständig mehrere Programme im Kernspeicher. Die Leer-

1) Martin F. WOLTERS, Der Schlüssel zum Computer, Textbuch 1, Düsseldorf 1969, S. 706

zeiten der Zentraleinheit werden weitgehend ausgeschaltet, indem dem Programm, das nach einer Anzahl ausgeführter Rechenoperationen auf eine Ein-/Ausgabeoperation wartet, vom Betriebssystem die Verfügbarkeit über die Zentraleinheit entzogen und einem anderen ablaufbereiten Programm übergeben wird. Muß dieses Programm nun seinerseits eine Ein-/Ausgabe vornehmen, kann entweder wieder das erste oder auch ein drittes Programm die Zentraleinheit übernehmen, usw. Haben die Programme die Ein-/Ausgabeoperationen durchgeführt, werden sie bei den meisten dieser Betriebssysteme in eine Warteschlange für die Zentraleinheitsbenutzung eingereiht, die von besonderen Routinen des Betriebssystems nach bestimmten Kriterien abgearbeitet wird. Die Multiprogrammverarbeitung zeichnet sich also dadurch aus, daß simultan eine fortlaufende Nutzung der Zentraleinheit stattfindet und die Programme, die keine Kontrolle über die Zentraleinheit haben, in der Zwischenzeit Lese- oder Schreiboperationen ausführen können. Dieses Verfahren kann zwar bei den einzelnen Programmen zu einer Erhöhung der Laufzeiten führen, hat aber insgesamt eine erhebliche Steigerung des Programmdurchsatzes zur Folge. Es muß allerdings darauf hingewiesen werden, daß auch die Zeit, die das Betriebssystem benötigt, um seine Funktionen wahrzunehmen, bei Multiprogramming-Systemen höher ist als bei Uniprogramming-Systemen. Dennoch läßt sich sagen, daß die Möglichkeiten eines Großrechners heute nur noch durch den Einsatz des Multiprogramming effizient genutzt werden können.

Die Verarbeitungsformen des Uniprogramming und des Multiprogramming sollen an Hand der Abbildungen 1 und 2 noch einmal erläutert werden.

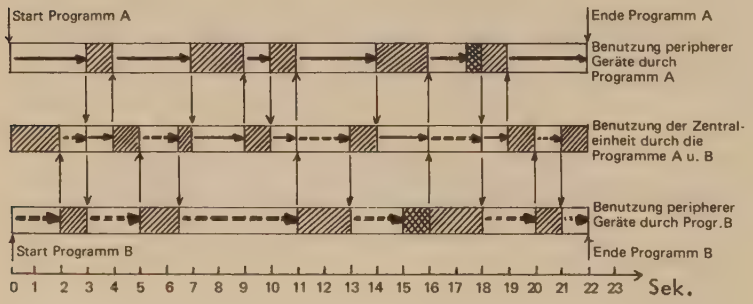
Abb. 1 : Ausnutzung der Zentraleinheit bei einer Verarbeitung im Uniprogramming



ungenutzte Zeiten der peripheren Geräte bzw. der Zentraleinheit

Zeiten, in denen die peripheren Geräte bzw. die Zentraleinheit beschäftigt sind.

Abb. 2 : Ausnutzung der Zentraleinheit bei einer Verarbeitung im Multiprogramming



Leerzeiten der peripheren Geräte bzw. der Zentraleinheit

Nutzung der peripheren Geräte bzw. der Zentraleinheit durch die Programme A (⟶) und B (- - →)

Wartezeit von Programm A oder B auf die Zuteilung der Zentraleinheit

Bei einer Verarbeitung im Uniprogramming führt der aus Abb. 1 ersichtliche ständige Wechsel zwischen der Benutzung der Peripherie und der Zentraleinheit durch das zu erledigende Programm zu einem sehr schlechten Ausnutzungsgrad der Zentraleinheit. Die in Abb. 2 vereinfacht dargestellte Multiprogrammverarbeitung für zwei voneinander unabhängige Programme A und B läßt dagegen erkennen, daß der Ausnutzungsgrad der Zentraleinheit bei dieser Verarbeitungsform wesentlich höher ist. Die Programme verlieren die Kontrolle über die Zentraleinheit jeweils bei einer Ein-/Ausgabeoperation. Nach Beendigung dieser Operationen werden sie wieder in die Warteschlange für die Benutzung der Zentraleinheit eingereiht, die nach bestimmten Regeln abgearbeitet wird. Die sich bei den einzelnen Programmen möglicherweise ergebenden mehrfachen Wartezeiten können gegenüber dem Uniprogramming zu einer Verlängerung der Laufzeiten der einzelnen Programme führen. In Abb. 2 treten diese Wartezeiten bei Programm A im Zeitraum 17,5 - 18 und bei Programm B im Zeitraum 15 - 16 auf. Die Laufzeit von Programm B würde sich gegenüber einer Erledigung im Uniprogramming beim Multiprogramming bei dieser Kombination mit Programm A um genau eine Zeiteinheit erhöhen, die Laufzeit von Programm A um 0,5 Zeiteinheiten. Bei Kombinationen mit anderen Programmen bzw. zeitlichen Verschiebungen können sich aber auch höhere oder niedrigere Laufzeiten ergeben.

2.2 Die Funktionen eines Betriebssystems

Ein Betriebssystem besteht aus einer Reihe von Programmen ("Systemprogramm-Software"), die vom Hersteller mitgeliefert werden. Es hat die primäre Aufgabe, den reibungslosen Ablauf der Programme innerhalb einer Datenverarbeitungsanlage zu gewährleisten und dem Benutzer ihre Möglichkeiten zu erschließen. Da eine moderne Datenverarbeitungsanlage ohne Betriebssystem nicht einsatzfähig ist, stellt dieses eine notwendige Ergänzung ihrer technischen Struktur (Hardware) dar.

Für ein Anlagenmodell bieten die Herstellerfirmen meist mehrere Betriebssysteme an, die sich durch den Bedienungskomfort, den Platz, den sie im Arbeitsspeicher belegen, sowie vor allem durch ihre Leistungsfähigkeit voneinander unterscheiden. Der Aufwand der Hersteller für diese Betriebssysteme darf nicht unterschätzt werden. Ihre Entwicklungskosten können die Kosten für die technische Entwicklung eines Rechnersystems durchaus überschreiten, und der Trend zu steigenden Kosten für die Entwicklung solcher Programme wird voraussichtlich in den nächsten Jahren noch zunehmen[1]. Eine genaue Abgrenzung zwischen der Hardware und der Software eines Rechnersystems wird in Zukunft jedoch immer schwieriger durchzuführen sein, da durch die Entwicklung der Mikroprogrammkreise manche Funktionen der Betriebssysteme, die herkömmlicherweise der Software zugerechnet werden, wieder als Hardware in die Rechner eingesetzt werden.

Je nach der Größe eines Rechners kann man zwischen verschiedenen Ausbaustufen der Betriebssysteme wählen. Man unterscheidet hauptsächlich drei Arten, nämlich Basisbetriebssysteme, Bandbetriebssysteme und Plattebetriebssysteme. Diese Betriebssysteme sind meist vom Basis- zum Plattebetriebssystem kompatibel, d.h. ein Programm, das man für das Basisbetriebssystem geschrieben hat, kann man auch im Band- bzw. im Plattebetriebssystem verwenden.

Ein Betriebssystem gliedert sich normalerweise in die drei Bestandteile
1. Kontroll- und Überwachungsprogramme (Organisationsprogramme),
2. Übersetzungsprogramme und 3. Dienstprogramme und Testhilfen.

Die Kontroll- und Überwachungsprogramme übernehmen viele Funktionen, die bisher vom Benutzer ausgeführt wurden. Sie sind für die Steuerung der Zentraleinheit, für die Verwaltung des Kernspeichers und sämtlicher angeschlossener Geräte verantwortlich. Einige Bestandteile dieser Programme müssen ständig im Kernspeicher stehen, andere werden

---

1) M.F. WOLTERS, a.a.O., S. 716

nur bei Bedarf von der System-Residenz [1] (Magnetband, Magnetplatte) in den Kernspeicher geholt. Zu den Funktionen dieser Organisationsprogramme gehören u.a.:

- das Zuteilen von Speicherplatz und externen Speichermedien,
- das Laden von Programmen und Programmsegmenten,
- die Programmsteuerung bei Multiprogramming,
- die Ablaufsteuerung mehrerer aufeinanderfolgender Programme ohne manuelle Eingriffe,
- die Vorrangsteuerung der Programme bei Angabe von Prioritäten,
- die Behandlung von Programmunterbrechungen,
- die Fehlerbehandlung bei Ein-/Ausgabeoperationen und
- die Definition und Verwaltung von Dateien.

Die Übersetzungsprogramme ermöglichen es dem Benutzer, eine von ihm ausgewählte, problemorientierte Sprache (FORTRAN, ALGOL, COBOL) in die Maschinensprache umzuwandeln.

In die Kategorie der Dienstprogramme und Testhilfen schließlich fallen alle Routinen, die dem Benutzer häufig wiederkehrende Arbeiten abnehmen, wie z.B. Sortieren und/oder Mischen von Datenbeständen, aber auch die Wartung von Programm- und Dateibibliotheken, das Überspielen von Datenbeständen, z.B. von einem Magnetband auf eine Magnetplatte, und das Testen neuerstellter Programme in Verbindung mit einer umfangreichen Fehlerdiagnostik.

Die in den folgenden Absätzen dargestellten Betriebssysteme sind sämtlich Plattebetriebssysteme, da nur sie das Multiprogramming erlauben.

---

1) Unter System-Residenz versteht man den physikalischen Datenträger, auf dem sich die Routinen des Betriebssystems befinden, die nicht ständig im Kernspeicher verfügbar sein müssen.

2.3 IBM: Operating System (OS) [1)]

Das Operating System (OS) der IBM ist eine Zusammenstellung von Verarbeitungsprogrammen (Sprachübersetzer und Service-Programme) sowie von einem integrierten Kontrollprogramm, das jede Arbeit, die von der Anlage erledigt wird, überwacht und koordiniert. Ziel des Betriebssystems ist es, dem Benutzer eine möglichst gute Nutzung der Anlage zu ermöglichen. Dazu ordnet beispielsweise das Kontrollprogramm unter anderem alle verfügbaren Systemeinheiten zu, wie z.B. die Zentraleinheit, den Kernspeicher, die Ein-/Ausgabe-Einheiten sowie alle im System befindlichen Programme (System- und Benutzerprogramme). Das Kontrollprogramm verwaltet außerdem die Ein- und Ausgabedatenbestände des Benutzers. Die Ausführungen dieses Abschnitts beschränken sich auf die Darstellung einzelner Elemente des Kontrollprogramms, da die in Kapitel 6 beschriebene Belegungsplanung von den Möglichkeiten des Benutzers zur Steuerung der Jobs ausgehen und dabei auch die interne Verwaltung der Jobs durch das Betriebssystem berücksichtigen muß. Auf die Sprachübersetzer und Service-Programme wird nicht eingegangen, obwohl sie für den Benutzer in der Regel wichtiger sind als das Kontrollprogramm selbst.

Die IBM bietet drei verschiedene Versionen des Kontrollprogramms an:

- das einfache Kontrollprogramm (Primary Control Program), welches die Erledigung nur jeweils eines Jobs überwacht,
- die Multiprogrammverarbeitung mit einer fest vorgegebenen Anzahl von Programmen (Multiprogramming with a Fixed Number of Tasks: MFT), bei der mehrere Programme zur gleichen Zeit überwacht werden. Die Abarbeitung der Programme erfolgt sequentiell in der gleichen Reihenfolge, in der sie eingelesen werden. Der Kernspeicher wird in dieser Version des OS bei der Systemgenerierung in gleich oder auch verschieden große Bereiche unterteilt, die man als "partitions"

---

1) IBM System /360 Operating System, Concepts und Facilities File No. S 360 - 20, Form C 28 - 6535 - 1, 2. Auflage 1967

bezeichnet. In jede partition kann ein Programm geladen werden, und diese Programme übernehmen die Zentraleinheit während ihrer Ausführung im Wechsel.

- die Multiprogrammverarbeitung mit einer variablen Anzahl von Programmen (Multiprogramming with a Variable Number of Tasks: MVT). Die MVT-Version überwacht ebenfalls mehr als einen Job im gleichen Zeitabschnitt, ordnet im Gegensatz zum MFT jedem Programm aber nur den Kernspeicherbereich zu, den es benötigt. In diesem Fall spricht man von sog. Kernspeicher-Regions. Die Reihenfolge der Programmerledigung erfolgt auf Grund von Prioritäten, die der Benutzer frei vergeben kann. Können zum gleichen Zeitpunkt mehrere Programme eingeplant werden, erhält stets das Programm mit der höheren Priorität die angeforderten Kapazitäten. Wie diese Vergabe im einzelnen erfolgt, wird noch erläutert.

Die drei dargestellten Versionen des Kontrollprogramms weichen vor allem in der Art der Einplanung der Programme und der Überwachung ihrer Ausführung voneinander ab. Den größten Spielraum für den Benutzer bietet dabei zweifellos die MVT-Version, deren Routinen nun näher erläutert werden sollen.

Die drei Hauptfunktionen des Kontrollprogramms sind folgende:

1. Annahme und Einplanung von Jobs in einem stetigen Prozeß (Job Management Routinen). Unter einem Job versteht man dabei eine benutzerbezogene Programmeinheit, die in der Regel aus mehreren voneinander abhängigen und nacheinander ablaufenden Einzelprogrammen (Job-Steps) besteht.

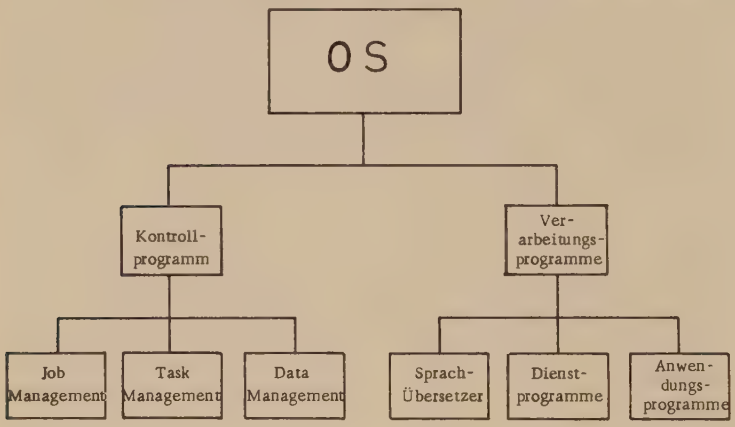
2. Überwachung jeder zu erledigenden Task nach Prioritätsgesichtspunkten (Task Management Routinen), wobei eine Task die maschinenbezogene Arbeitseinheit ist, die vom System erledigt werden muß. Die Task entspricht dem einzelnen Job-Step, und zwar wird der Job-Step in dem Moment zur Task, in dem die Job Management

Routinen die Kontrolle über die Ausführung des Programms an die Task Management Routinen übergeben.

3. Steuerung aller Operationen, die mit Ein-/Ausgabe-Einheiten [1] in Verbindung stehen (Data Management Routinen).

Den Aufbau des Operating Systems gibt Abb. 3 wieder:

Abb. 3: Aufbau des Operating Systems (OS)



Die Programme des Job Management steuern und kontrollieren den Ablauf eines oder mehrerer Jobs. Dabei dienen Steuerkarten (sog. Job-Control-Statements) als Verbindungselemente zwischen den Benutzerprogrammen und dem System. Die Folge der Job-Control-Statements bezeichnet man als Eingabestrom (input stream). Die Aufgaben des Job-Managements sind im einzelnen:

1. Die Analyse des Eingabestroms: Abfrage der Eingabedaten, um die Control-Statements zu identifizieren; Analyse und Interpretation der Control-Statements; Vorbereitung bestimmter Kontroll-Tabellen, durch die jeder Job dem System genau beschrieben wird.

---

1) Abgekürzt soll im folgenden der Ausdruck E-/A-Einheiten verwendet werden.

2. Die Zuordnung von E-/A-Einheiten: Kontrolle, ob alle benötigten Ein-/Ausgabeeinheiten tatsächlich zugeordnet sind; Kontrolle, ob der Speicherplatz zugeordnet worden ist, der für Dateien benötigt wird, die im direkten Zugriff stehen; Kontrolle, ob der Operator alle verlangten Magnetbänder und -platten montiert hat.

3. Gesamtplanung: Auswahl von Jobs für die Ausführung. Die Gesamtplanung wird - in Abhängigkeit von dem gewählten Betriebssystem - auf sequentieller Basis oder nach bestimmten Prioritätsregeln durchgeführt.

4. Übertragung von Eingabedaten auf Dateien bzw. von Ausgabedaten von Dateien, die im direkten Zugriff stehen.

5. Kommunikation zwischen dem Operator und dem System.

Der Benutzer teilt dem System über die Job-Kontroll-Sprache die notwendigen Informationen über seinen Job und die entsprechenden Job-Steps mit. Insbesondere kommen dafür 6 JCL-Statements in Frage:

1. Job Statement

   Es markiert den Beginn eines Jobs und im Input Stream das Ende der Control-Statements des Vorjobs.

   Beispiel einer Job-Karte:

   J100000 JOB (002,519),MUELLER,CLASS=B,PRTY=12,REGION=100K

   Es bedeuten dabei:

   J 100000 - Name des Jobs

   002,519 - Abrechnungsinformationen über den Job

   MUELLER - Name des Programmierers

   CLASS=B - Angabe der Jobklasse, zu der ein Job gehört

   PRTY=12 - die für den Job festgelegte Priorität. Die Priorität kann zwischen 0 und 14 definiert werden. "14" - als höchste Priorität - sollte man dabei möglichst nicht wählen, da mit dieser Priorität die System-Programme abgewickelt werden. Die Angabe der Priorität kann auch unterbleiben, da bei der Systemgenerierung eine Standardpriorität für alle Jobs vordefiniert wird.

100 K - die Angabe des für den Job erforderlichen Kernspeicherplatzes in Kilo-Bytes. Sie muß sich an dem Bedarf des größten Job-Steps orientieren. Soll die Angabe stepbezogen erfolgen, muß sie in der EXEC-Karte stehen.

2. Execute Statement (EXEC-Statement)

Das EXEC-Statement markiert den Beginn eines Job-Steps und identifiziert das erste Programm, das ausgeführt werden muß.

Beispiel: STEP 1 EXEC PGM=S 099092

Durch dieses Statement wird das Programm 099092 in den Kernspeicher geladen.

3. Datei-Definitions Statement (DD-Statement)

Das DD-Statement beschreibt die Datenbestände, die in einem Job erstellt oder benutzt werden. Von den zahlreichen Merkmalen, die der Benutzer in diesem Statement verschlüsseln kann, seien nur folgende genannt:

- Name der Datei
- Typ und Anzahl der für diese Datei bestimmten E-/A-Einheiten (Magnetbänder, Magnetplatten)
- Datenträger, auf dem sich die Datei befindet (z.B. Rollen-Nummer des Magnetbandes)
- Menge des für eine Datei auf einem Datenträger mit direktem Zugriff reservierten Platzes (Anzahl der Sätze, Spuren oder Zylinder)
- Disposition der Datei (NEW: neu zu erstellen, OLD: bereits bestehend, PASS: weitergeben an den nächsten Jobstep, DELETE: löschen, CATALOG: katalogisieren, usw.)
- Zugriffsmethode (sequentiell, index-sequentiell, direkt)
- Kennsatzinformationen
- Satz- und Blocklänge.

4. Command Statement

Das Command Statement wird vom Operator benutzt, um dem System Befehle im Eingabestrom mitzuteilen. Die Anweisungen können Ein- und Ausgabeeinheiten aktivieren bzw. deaktivieren und eine Anzahl anderer

Funktionen durchführen, die der Operator sonst von der Konsole her eingibt. Sie entlasten den Operator also bei vorhersehbaren Änderungen in der System-Konfigurierung.

5. Delimiter Statement

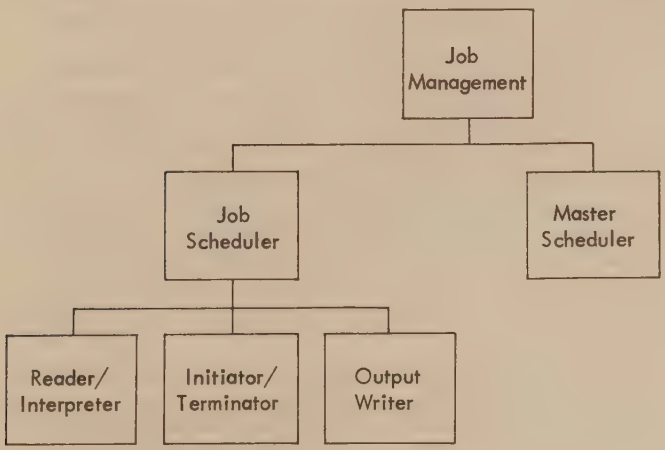
Das Delimiter Statement stellt eine Markierung im Eingabestrom dar, d.h. es wird benutzt, wenn Daten an einer bestimmten Stelle in den Eingabestrom eingefügt werden sollen. Durch dieses Statement werden die Datenkarten von den anderen Steuerkarten getrennt.

6. Null Statement

Das Null Statement wird benötigt, wenn das letzte DD-Statement einen Datenbestand definiert hat, der im Input-Stream liegt. Es zeigt das Ende eines Jobs an.

Diese kurzen Ausführungen über die Job-Control-Language haben die Möglichkeiten des Benutzers dargestellt, auf das Job-Management des Betriebssystems Einfluß zu nehmen. Die Planung und Steuerung des eigentlichen Verarbeitungsablaufs erfolgt dann mit Hilfe von zwei Hauptroutinen, nämlich des Job Schedulers und des Master Schedulers, die Abb. 4 schematisiert zeigt:

Abb. 4: Elemente des Job Management



Der Job Scheduler bereitet Jobs für die Ausführung vor und plant ihre Ausführung entweder auf Prioritäts- oder auf sequentieller Basis. Er hat drei Aufgaben innerhalb des Kontroll-Programms zu erledigen, nämlich 1. die Steuerkarten zu lesen und zu interpretieren, 2. Jobs und Job-Steps zu initialisieren sowie zu terminieren und 3. die Aufbereitung der Ausgabe zu überwachen.

Zum Lesen und Interpretieren der Job-Control-Statements wird vom Operator nach der Systemgenerierung ein sog. Reader/Interpreter gestartet. Er liest die Job-Control-Statements des Eingabestroms, analysiert ihren Inhalt und baut eine Reihe von Tabellen auf, die bei der Initialisierung und Ausführung eines Job-Steps benötigt werden. Bei einem Prioritäts-Planungs-System reiht er die Jobs ihrer Priorität entsprechend in eine Warteschlange ein und überträgt zusätzlich die Daten, die im Eingabestrom enthalten sind, auf Dateien im direkten Zugriff.

Die Initialisierung und Terminierung [1] eines Jobs (Job-Steps) wird von einer Initiator/Terminator-Routine abgewickelt, die nach der Beendigung eines Job-Steps den nächsten Job-Step auswählt, der zur Ausführung gelangen soll, und zwar entweder aus dem Eingabestrom bei sequentieller Verarbeitung oder aus einer Warteschlange (input work queue) bei prioritätsgesteuerter Verarbeitung.

Exkurs: Festlegung der Bearbeitungsreihenfolge von Jobs im OS

Die Bearbeitungsreihenfolge der Jobs kann im OS nicht nur mit Hilfe der Jobpriorität festgelegt werden, sondern man hat die zusätzliche Möglichkeit, nach beliebigen Kriterien sog. Jobklassen zu bilden. Diese Jobklassen können mit den 15 Buchstaben A bis O verschlüsselt werden. Man kann sie beispielsweise nach der Anzahl der maximal in einem Job benutzten Bandeinheiten vergeben oder der Klasse A allgemeine Jobs, B Jobs mit Band-Dateien, C rechenintensive Jobs, E ein-/ausgabeintensive Jobs, usw. zuordnen. Die Festlegung der für einen Job zutreffenden Jobklasse erfolgt im Job Statement[2]. Der Anzahl der Jobklassen entsprechend können in der MVT-Version des OS bis zu 15 verschiedene Initiator/Terminator gestartet werden, die jeweils maximal acht Jobklassen bedienen können. Ver-

1) Der Initiator ist für die Auswahl des nächsten zu erledigenden Job-Steps und für die vorbereitenden Arbeiten verantwortlich, die zu seiner Ausführung notwendig sind, der Terminator entfernt die zu einem Step gespeicherten Daten aus den Tabellen des Kontrollprogramms.

2) Dies ist natürlich nur erforderlich, wenn man von der Möglichkeit Jobklassen zu definieren, Gebrauch gemacht hat.

gibt man für den ersten Initiator die Klassen A, B, C und für den zweiten die Klassen B, C, A, D, so durchsucht der erste Initiator den Input-stream zunächst nach einem Job der Klasse A, dann nach B, dann nach C; der zweite beginnt mit B und sucht dann nach C, A und D. Findet er einen Job der Klasse, für die er zuständig ist, initialisiert er den ersten Step dieses Jobs. Erst wenn er mehrere Jobs der gleichen Klasse findet, wählt er den Job mit der höchsten Priorität aus. Die Angabe der Job-Priorität hat bei der Angabe von Jobklassen also nur eine sekundäre Bedeutung. Macht man dagegen keinen Gebrauch von den Jobklassen, erfolgt die Auswahl ausschließlich auf Grund der Prioritäten.

Der Initiator/Terminator analysiert weiterhin die Anforderungen hinsichtlich benötigter E-/A-Einheiten, ordnet ihnen verfügbare Einheiten zu, schreibt auf der Konsolschreibmaschine die entsprechenden Hinweise zum Montieren der Bänder, prüft, ob die eingespannten Bänder auf der richtigen Einheit montiert sind und führt abschließende Arbeiten bei Beendigung des Jobs oder Jobsteps aus (z.B. Freigabe der benötigten E-/A-Einheiten).

Während der Ausführung einer Task in einem System, das mit Hilfe von Prioritäten geplant wird, können Ausgabedaten zunächst auf Magnetplattenspeicherbereichen gespeichert werden. Später können sog. "output writer" diese Daten auf vordefinierte System-Ausgabegeräte übertragen (Druck- oder Stanzeinheiten). Dabei können die einzelnen Ausgabeeinheiten in verschiedene Ausgabeklassen unterteilt werden, um möglichst schnell die Ergebnisse wichtiger Arbeiten ausgedruckt zu bekommen. Der Job Scheduler baut also sowohl für die Eingabe als auch für die Ausgabe Warteschlangen auf. Auch die Ausgabe-Warteschlangen sind dabei innerhalb der gleichen Klassen nach Jobprioritäten geordnet. Während ein Jobstep von Job A ausgeführt wird, können von Job B, der bereits beendet wurde, also noch Ergebnisse ausgedruckt werden. Diese Eigenschaft hochentwickelter Betriebssysteme ermöglicht einen außerordentlich hohen Jobdurchsatz für eine gegebene Zeitspanne.

Der Master Scheduler hat die Aufgabe, Operator-Befehle und -Nachrichten zu behandeln. Die Nachrichten können durch einen Befehl vom Benutzer-Programm an den Operator weitergegeben werden, und der Operator kann diese Nachricht durch eine Antwort an das Programm zurück-

geben. Außerdem kann er selbst verschiedene Befehle und Anfragen an das System geben:

- Befehle, die sich auf den Status eines Jobs beziehen (Job Action Commands)
  Er kann einen Job canceln, d.h. seine Ausführung abbrechen, ihn in eine Warteschlange einreihen oder seine Priorität (aus dem JOB-Statement) ändern, solange er noch nicht initialisiert worden ist.
- Befehle, die sich auf das System beziehen (System Action Commands)
  Diese Befehle bewirken Änderungen für den Job Scheduler. Sie können zum Beispiel eine neue Einheit für den Eingabestrom definieren, bestimmte Magnetplattenspeicher als nicht mehr zuordnungsbar erklären oder einem Drucker eine andere Druckausgabeklasse zuweisen.
- Anfragen an das System (Information Requests)
  Sie gestatten es dem Operator, sich jederzeit ein vollständiges Bild über den Status des Systems und auch einzelner Jobs zu verschaffen.

Diese Operator-Befehle können über eine Konsolschreibmaschine dem System direkt eingegeben werden, sie können aber auch als Statements in den Eingabestrom eingefügt werden.

Aufbauend auf diesen Angaben über das Job Management, kann jetzt die zweite Gruppe der Kontrollprogramme, die Task Management-Routinen, erläutert werden. Sie überwachen die Ausführung aller Arbeiten, die im System vorgenommen werden. Es muß also jede Arbeit, die dem System zur Ausführung übertragen wird, als Task formuliert werden; unter einer Task versteht man dabei die kleinste Arbeitseinheit, die sich um die Benutzung der Zentraleinheit, des Kernspeichers und der E-/A-Kanäle bemühen kann.

Bei einer Maschinenausstattung, die das Abarbeiten mehrerer Tasks erlaubt (multitasking environment), hängt die Reihenfolge, in der die Jobs zur Ausführung ausgewählt werden und die Reihenfolge der Kapazitätszuordnung

von den vergebenen Prioritäten ab. Diese Prioritätsvergabe kann einmal im JOB-Statement erfolgen, sie kann aber auch vom Operator durchgeführt werden, solange der Job nicht tatsächlich zur Ausführung ausgewählt wird. Initialisiert der Job Scheduler einen Step, wird die Jobpriorität systemintern zur Festsetzung einer sog. "Dispatching Priority" benutzt, die für die Zuordnung der Zentraleinheit und des Kernspeichers maßgebend ist.

Jede System-Kapazität besitzt dazu ihren eigenen "Manager". Es gibt also einen Manger für die Zuordnung der Zentraleinheit (Task Dispatcher), einen für die Kernspeicherzuteilung und einen für die E-/A-Kanalverteilung. Diese Manager bauen sich nach Prioritäten geordnete Warteschlangen auf, um einen Überblick über unbefriedigte Zuordnungswünsche zu haben und vergleichen diese mit Tabellen der verfügbaren Kapazitäten.

Die Task Management Programme gewährleisten jederzeit einen vollständigen Überblick über alle Zuordnungen und stellen sicher, daß bei Beendigung einer Task die genannten Systemkapazitäten wieder freigemacht werden. Dazu benutzt das Task Management sog. "Task Control Blocks (TCB)", die alle Informationen enthalten, die sich auf eine Task beziehen. Dazu gehören Registerinhalte sowie Kernspeicherbereiche, die der Task zur Verfügung gestellt wurden. Die TCB werden aufgebaut, wenn die Task entsteht, d.h. wenn ein Job-Step von einem Initiator des Job Schedulers initialisiert wird.

Der Status einer Task hängt von ihrer Bereitschaft ab, die Zentraleinheit (CPU) zu benutzen. Wenn sie augenblicklich in der Lage ist, die CPU zu benutzen, besitzt sie den Status "ready". Wenn sie die CPU benutzt, ist sie "active" und anderenfalls ist sie im "wait"-Zustand. In den "wait"-Zustand kann die Task z.B. gelangen, wenn sie auf die Zuteilung eines E-/A-Kanals wartet, oder wenn sie liest bzw. schreibt.

Obwohl die Forderung, daß jede Arbeit unter der Kontrolle der Task Management Programme abläuft, keine Ausnahme zuläßt, kann die Art und Weise der Kontrolle je nach der Wahl der Verarbeitungsform stark variieren. In

der einfachsten Form wird immer nur ein Programm erledigt, und dieses Programm kann über alle Kapazitäten verfügen. Bei dieser sog. "single task environment" können keine Engpässe auftreten, da die Jobsteps einzeln sequentiell erledigt werden. Die MFT-Version des Operating Systems erlaubt die Multiprogrammverarbeitung und führt so zu einer wesentlich besseren Ausnutzung der Zentraleinheit. Die Anzahl der Tasks, die in den Kernspeicher geladen werden können, liegt durch die Anzahl der definierten Partitions fest.

Interessant ist, daß die Jobklassen in dieser Version nicht den Initiatoren zugeordnet werden, sondern den einzelnen Partitions und daß die Initiatoren jeweils eine bestimmte Partition bedienen. Es können also nur so viele Initiatoren gestartet werden, wie man Partitions definiert hat.

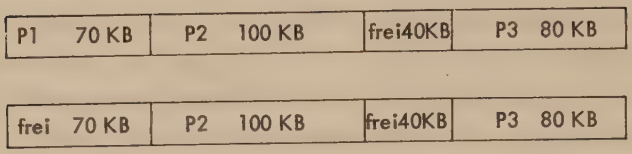
Die maximal mögliche Zahl von Partitions ist 15. Das MFT ordnet stets der Partition P 0 die höchste Dispatching-Priorität zu, d.h. die Jobklassen, die man für diese Partition definiert, werden bei der Zuteilung des Kernspeichers und der Zentraleinheit bevorzugt bedient. Die Partition P 1 und damit auch die ihr zugehörigen Jobklassen erhalten eine niedrigere Dispatching-Priorität usw. Die im Job Statement definierte Prioritätsangabe bezieht sich bei dieser Version des OS nur auf die Stellung des Jobs innerhalb des Eingabestroms, ist aber nicht - wie im MVT - mitbestimmend für die Dispatching-Priorität. Hat man der Partition P 0 die Klassen A und B zugeordnet und der Partition P 1 die Klassen B, C und A, und geht man weiter davon aus, daß die Klasse A für sehr eilige Jobs reserviert ist, so kann es vorkommen, daß bei diesem System ein Job der Klasse A dennoch sehr lange auf seine Erledigung warten muß, wenn in P 0 gerade ein sehr langlaufender Job der Klasse B initialisiert ist, der den Klasse-A-Job, der in Partition P 1 läuft, immer wieder unterbricht, weil die Partition P 0 Priorität genießt.

Der Benutzer hat bei diesem System kaum eine Möglichkeit, die Reihenfolge der Jobs zu beeinflussen, es sei denn, er besitzt bei der Festlegung der Jobklassen für die einzelnen Partitions sehr genaue Informationen über

die Dringlichkeit seiner Jobs und ordnet dementsprechend den Partitions die Jobklassen zu. Diesen Nachteil des MFT besitzt die am weitesten entwikkelte Version des Operating Systems, das "MVT", nicht. Hier ist die Prioritätsvergabe im JOB-Statement auch für die Festlegung der Dispatching-Priorität maßgebend. Durch eine entsprechende Definition der Jobklassen, die beim MVT den Initiatoren zugeordnet werden, kann der Benutzer hier direkt auf die Reihenfolge Einfluß nehmen, in der seine Jobs ablaufen. Ordnet man z.B. jedem Initiator als erstes die Klasse A zu, so kann man über eine hohe Prioritätsangabe im Job-Statement sicherstellen, daß der eilige Job vom nächsten freien Initiator angenommen wird und dann auch tatsächlich Priorität vor allen anderen Jobs hat. Außerdem ist in dieser Version die Zahl der parallel ablaufenden Programme variabel, da jedes Programm nur soviel Kernspeicher belegt wie es auch tatsächlich benötigt. Die übrigen Eigenschaften des MVT entsprechen weitgehend denen des MFT, d.h. es können max. 15 Jobs parallel laufen, die von max. 15 Initiatoren gesteuert werden. Außerdem können pro Initiator - wie beschrieben - 8 Jobklassen vergeben werden, die von den Initiatoren in der angegebenen Reihenfolge erledigt werden. Obwohl dieses System wesentlich mehr Komfort bietet als die rein sequentielle Verarbeitung und auch die Verarbeitung in festen Partitions, ist es in zwei Punkten noch nicht ausgereift:

1. Eine laufende Task kann nicht unterbrochen werden und das bedeutet, daß keine Neuordnung des Kernspeichers nach dem Ende eines Jobs bzw. Job-Steps vorgenommen werden kann, die aber erforderlich wäre, um wieder zusammenhängende freie Kernspeicherbereiche zu schaffen. Diese Problematik des core fregmentation soll an Hand des in Abb. 5 dargestellten Beispiels kurz erläutert werden. Ist das Programm P 1, das 70 KB benötigt, beendet worden und greift der frei gewordene Initiator das Programm P 4 aus dem Eingabestrom heraus, das 90 KB verlangt, dann kommt dieses Programm nicht zum Zuge, selbst wenn an einer anderen Stelle des Kernspeichers noch eine Region von 40 KB frei ist, insgesamt also 110 KB verfügbar sind, denn die in der Ausführung befindlichen Programme P 2 und P 3 können nicht zusammengeschoben werden.

Abb. 5: Problematik des Core fregmentation



In die freien Bereiche können also stets nur Programme in der Größe der jeweils freien Region eingefügt werden. Der Initiator, der das 90 KB Programm ausgewählt hat, bleibt aber belegt, bis eine entsprechende Region frei wird. Das Programm P 4 kann erst ablaufen, wenn P 2 oder P 3 endet, da dann eine zusammenhängende Region von insgesamt 210 KB bzw. 120 KB verfügbar ist.
Solange IBM dieses Problem noch nicht zufriedenstellend gelöst hat, kann man sich helfen, indem man die Regions bei der Systemgenerierung so definiert, daß der größte Jobstep ohne Wartezeit erledigt werden kann. Man benutzt dann also eine Kombination zwischen MFT und MVT, indem man einmal feste Regions definiert, auf der anderen Seite aber den Vorteil einer prioritätsgesteuerten Verarbeitung ausnutzt.

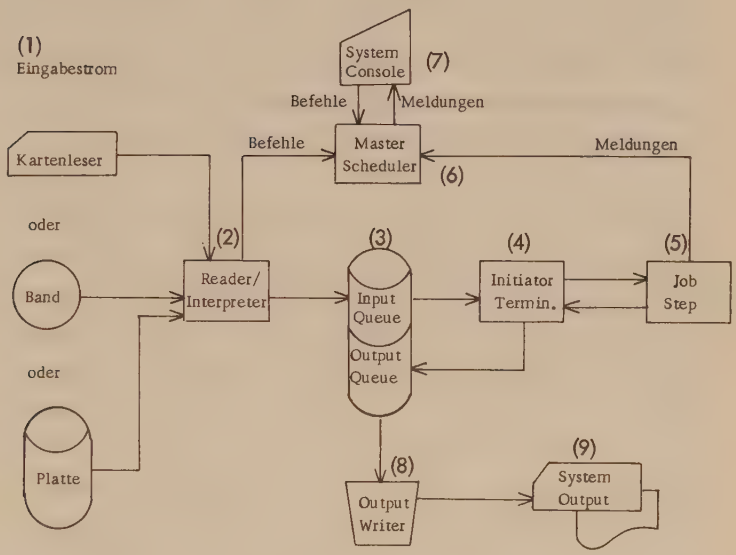
2. Die Zuordnung der Kapazitäten erfolgt immer in der Reihenfolge

    1. Belegung einer freien Region und
    2. Zuordnung der verlangten Einheiten.

Bei einem genügend großen Kernspeicher kann es dabei vorkommen, daß zwar eine freie Region belegt wird, daß es aber nicht möglich ist, auch die entsprechenden peripheren Geräte zuzuordnen. In diesem Fall wäre eine Auslegung des Betriebssystems wünschenswert, die die Initialisierung eines Jobsteps erst dann zuläßt, wenn beide Kapazitäten frei sind.

Abb. 6 zeigt abschließend noch einmal die Verknüpfungen zwischen dem Job Management und dem Task Management.

Abb. 6: Ein prioritätsgesteuertes Planungssystem[1)]



Die Steueranweisungen und Definition der Jobs und Jobsteps bilden den Eingabestrom, der von einer Karten-, Band- oder Platteneinheit gelesen werden kann (1). Der Reader/Interpreter liest und analysiert die Steueranweisungen für einen oder mehrere Jobs (2) und bringt sie in die Eingabewarteschlange. Dort werden sie nach Jobklassen und innerhalb dieser nach Prioritäten geordnet (3). Der Job mit der jeweils höchsten Priorität wird vom Initiator/ Terminator zur Ausführung ausgewählt (4) und unter der Steuerung des Task Management ausgeführt (5). Der Master Scheduler verarbeitet Operator-Befehle aus dem Eingabestrom und vom Konsol und gibt Nachrichten auf dem Konsol aus (6) und (7). Die Ausgabe von Daten aus der Ausgabe-Warteschlange erfolgt nach der Beendigung des Jobs durch den Output Writer gleichzeitig mit der Duchführung anderer Jobs (8) und (9).

1) IBM, System /360 Operating System, Concepts and Facilities, a.a.O., S. 40

Der dritte große Block neben 1. den Job Management- und 2. den Task Management-Routinen innerhalb der Kontroll- und Überwachungsprogramme befaßt sich mit der Dateiverwaltung. Die hierzu verwendeten DATA MANAGEMENT-Routinen des OS sollen allerdings nur insoweit dargestellt werden, wie dies für das weitere Verständnis der Arbeit notwendig erscheint. Auf Fragen der Speicherungsform, der Datensicherung und der Satzarten kann in diesem Zusammenhang nicht eingegangen werden.

Die Data Management-Programme sind primär für die Informationsübertragung zwischen Kernspeicher und peripheren Speichermedien verantwortlich und sind für die Datenspeicherung, die Vorbereitung von Kernspeicherbereichen für die Datenaufnahme sowie für das Lesen und Schreiben von Daten entwickelt worden. Um die Datenbestände des Benutzers verarbeiten zu können, benötigt das System Informationen über sie. Die meisten Angaben werden in dem bereits beschriebenen Datei-Definitions-Statement der Job-Control-Sprache festgelegt. Die Data Management-Routinen speichern diese Datei-Beschreibungen in sog. "Data Control Blocks" (DCB's), die bei der Ausführung eines Programms im Kernspeicher aufgebaut werden. In der Zeit, in der die Datenbestände nicht benutzt werden, sind die mit ihnen assoziierten Informationen auf einer System-Datei gespeichert, die als Katalog bezeichnet wird. Auf Grund des Datenbestandnamens findet das System mit Hilfe dieses Katalogs jederzeit alle für einen bereits existierenden Datenbestand gespeicherten Informationen wieder.

Die Routinen des Data Management unterstützen in besonderem Maß die manuell nur mit großem Aufwand mögliche Führung von Datenbeständen, die in periodischen Zeitabständen verändert werden. Jede Fortschreibung (Update) eines so definierten Bestandes bezeichnet man als Generation, und die Nummer, die das System für jede Generation vergibt, ist die sog. Generations-Nummer. Datenbestände, die in verschiedenen Generationen geführt werden, können mit Hilfe ihres Namens und der entsprechenden Generations-Nummer gefunden werden. Auf eine bestimmte Generation kann man sich dabei entweder durch die Angabe der absoluten oder einer relativen Generations-Nummer beziehen. Bezeichnet man z.B. eine

Datei mit dem Namen F 10000 und hat man von dieser Datei bisher 25 Generationen erstellt, von denen immer die letzten drei aufbewahrt werden, so hat diese Datei im Katalog den Dateinamen F 10000 G 0025, die vorletzte hat den Namen F 10000 G 0024 usw. Die jeweils zuletzt erstellte Datei wird vom System aber ebenfalls unter einer relativen Generationsnummer F 10000 (0) gefunden, die vorletzte unter F 10000 (-1) usw. Macht man in einem DD-Statement die Angabe F 10000 (+1), so bedeutet das, daß der entsprechende Step von der Datei F 10000 die Generation 26 erstellt, die dann beim nächsten Lauf dieses Steps wieder als F 10000 G 0026 oder F 10000 (0) angefordert werden kann. Der Benutzer kann dem Katalog dabei vorgeben, wieviele Generationen aufbewahrt werden sollen. Wird dann eine neue Generation erstellt, fällt die älteste automatisch aus dem Katalog heraus und der entsprechende Datenträger kann wieder anderweitig verwendet werden.

Diese Ausführungen über einige Routinen des Operating Systems betreffen nur einen kleinen Ausschnitt aus dem komplexen Kontrollprogramm. Sie sollten vor allem deutlich machen, daß das Betriebssystem nicht in der Lage ist, die Abwicklung der Benutzer-Programme unter optimierenden Gesichtspunkten vorzunehmen. Über die Jobklasseneinteilung und Prioritätsvergabe überläßt es vielmehr dem B e n u t z e r die Steuerung.

Dieser Abriß sollte aber auch deutlich machen, daß sich die Steuerung durch den Benutzer auf eine relativ globale Belegungs-Planung der peripheren Einheiten und des Kernspeichers beschränken muß. Den Jobablauf in der Anlage selbst sollte der Benutzer nicht zu beeinflussen versuchen, da dies nur über Eingriffe in das Betriebssystem erfolgen könnte und seine Möglichkeiten damit bei weitem überfordert wären.

2.4 UNIVAC: Executive System (EXEC 8) [1]

Das Betriebssystem EXEC 8 wurde von Univac für die Computer-Serie 1100 entwickelt und sieht einen Multiprogramming-Ablauf vor, der durch extern vorgebbare und intern berechnete Prioritäten gesteuert wird. In diesem System sind drei Programmstrukturen möglich:

1. Die Real-Time-Verarbeitung, bei der die Daten unmittelbar nach ihrem Entstehen verarbeitet und die Ergebnisse sofort ausgegeben werden. Programme dieser Kategorie besitzen die höchste Prioritätsstufe innerhalb des Betriebssystems und können vom Operator nicht unterbrochen werden.

2. Die Time-Sharing-Verarbeitung, bei der eine Vielzahl von Benutzern über Terminals die Bearbeitung von Programmen einschließlich der Ein-/Ausgabe anfordern kann. Jeder Benutzer erhält eine Zeitscheibe zugeteilt, deren minimale und maximale Größe bei der Systemgenerierung festgelegt werden. Es handelt sich bei Programmen dieser Art meist um solche für rechenintensive Probleme mit einem geringen Datenanfall.

3. Die Stapelverarbeitung (Batch-Verarbeitung), bei der das anfallende Datenmaterial zunächst gesammelt und dann in einem Schub mit Hilfe eines Programms oder einer Folge von Programmen verarbeitet wird. Der größte Teil der heute auf EDV-Anlagen abgewickelten kommerziellen Aufgaben wird in Stapelverarbeitung durchgeführt. Da sich die Ausführungen über das Operating System der IBM auf diese Verarbeitungsform bezogen haben, soll auch bei der Darstellung von EXEC 8 nur auf die Eigenschaften des Systems eingegangen werden, die sich auf die Batch-Verarbeitung beziehen. Im Unterschied zum OS erhalten auch Batch-Programme eine bei der Systemgenerierung definierte Zeitscheibe zur Benutzung der Zentraleinheit zugeteilt. Überschreiten sie das durch diese Zeitscheibe festgelegte Zeitintervall, wird ihnen die Kontrolle über die CPU entzogen. Sie verlieren die Kontrolle weiterhin, wenn sie E-/A-Operationen durchführen.

---

1) UNIVAC 1108 Multi Processor System, Operating System EXEC 8, Programmers Reference, UP-4144 Rev. 1, 1968

Auch beim Betriebssystem EXEC 8 setzt sich der Benutzer über Steuerkarten und eine Kontrollsprache (Executive Control Language) mit dem Rechner in Verbindung. Das für diese Arbeit wichtigste Statement ist das Run Statement, das mit dem Job Statement des OS vergleichbar ist. Es markiert den Beginn einer Arbeitsfolge, liefert Informationen zur Identifikation des Jobs und ermöglicht seine Eingliederung in eine Warteschlange nach Prioritätsgesichtspunkten. Die folgenden acht Einzelangaben sind im Run Statement möglich:

1. Priorität

   Die erlaubten Prioritäten können von A - Z festgelegt werden. Dabei ist A die höchste, Z die niedrigste externe Priorität.

2. Optionen hinsichtlich der Jobterminierung

   Folgende Optionen sind möglich:

   T = Terminieren, wenn die geschätzte Laufzeit überschritten wurde.

   P = Terminieren, wenn eine vorgegebene Anzahl Druckseiten erreicht ist.

   C = Terminieren, wenn mehr als eine bestimmte Anzahl Karten gestanzt wurden.

   S = Der Job darf erst gestartet werden, wenn der Vorjob beendet ist, der im Eingabestrom vor ihm liegt.

3. Run-Identifikation

   Erlaubt ist eine 6-stellige alphanumerische Bezeichnung.

4. Kostenstelle

   Bei jedem Job muß das Kostenstellenfeld ausgefüllt sein. Die möglichen Kostenstellen werden bei der Systemgenerierung mit angegeben; enthält ein Job eine ungültige Kostenstelle, wird er vom Rechner nicht ausgeführt.

5. Projektname

   Dieses Feld gibt dem Benutzer die Möglichkeit, einen weiteren 12-stelligen alphanumerischen Projektnamen anzugeben, z.B. für die Logbuch-Routinen, die vom Rechner ausgeführt werden.

6. Laufzeit/Deadline

   Dieses Feld kann wahlweise ausgefüllt werden; es vermittelt dem System zusätzliche Einplanungsinformationen.

Bei der Laufzeit kann der Programmierer die voraussichtlich benötigte CPU-Zeit in Minuten angeben. Überschreitet der Lauf diese Zeit, wird er entweder augenblicklich beendet oder das System überläßt dem Operator die Entscheidung über die Terminierung des Jobs.

Die Deadline-Angabe stellt eine Besonderheit des EXEC 8 dar, die es beim OS in dieser Form nicht gibt. Hier kann der Programmierer spezifizieren, bis zu welcher Uhrzeit ein Lauf spätestens beendet sein muß, bzw. welche Zeit von der Eingabe bis zur Fertigstellung höchstens verstreichen darf. Macht der Programmierer die Deadline-Angabe, muß er selbstverständlich auch die geschätzte Laufzeit angeben.

Man muß sich bei der Ausnutzung dieser Option darüber im klaren sein, daß sie die normale Prioritätssteuerung des Systems empfindlich stört, und zwar vor allem, wenn der "Deadline-Job" hinsichtlich seines Fertigstellungstermins kritisch wird. Er bekommt intern dann eine so hohe Priorität, daß praktisch kein anderer Job mehr über die Zentraleinheit verfügen kann. Verfolgt man das Ziel eines möglichst großen Jobdurchsatzes, können die "Deadline-Jobs" zu sehr großen Verzögerungen bei den anderen Jobs führen.

7. Angabe der maximal erlaubten Seitenzahl und der maximal abzustanzenden Anzahl der Karten

Wird die hier spezifizierte Seitenzahl bzw. Kartenanzahl überschritten, beendet das System den Job, wenn man die entsprechende Option für die Jobterminierung benutzt hat.

8. Startzeit

Auch die Möglichkeit, eine Startzeit zu definieren, zu der der Job frühestens beginnen kann, kennt nur das EXEC 8. Die Angabe einer Startzeit kann sich als nützlich erweisen, wenn Daten erst zu einem bestimmten Zeitpunkt verfügbar sind oder wenn man wünscht, daß bestimmte Produktionsplanungs- oder Systemprogramme nicht vor einem bestimmten Zeitpunkt laufen. Jobs mit Startzeit werden erst dann mit ihrer Priorität in die Warteschlange eingereiht, wenn der definierte Zeitpunkt überschritten ist. Auch wenn gleichzeitig die Deadline-Angabe erfolgte, wird der Job erst interpretiert, wenn diese erreicht ist. Die Startzeit wirkt wie der Zeitpunkt der Eingabe der Job-Steuerkarten.

Beispiel für den möglichen Aufbau eines RUN-Statements:

@ RUN,E/TCS,J1,3115,PLANUNG,10/250,/150,D930
1 2 3 4 5 6 7 8

1: Die Job-Priorität ist E.

2: Der Job darf erst laufen, wenn der Vorjob abgeschlossen ist. Er wird abgebrochen, wenn er mehr als 10 Min. (6) CPU-Zeit benötigt oder mehr als 150 Karten (8) abstanzt.

3: Die Identifikationsnummer des Runs ist J1.

4: Die belastete Kostenstelle trägt die Nummer 3115.

5: Der Job befaßt sich mit dem Projekt "Planung".

6: Der Job darf höchstens 10 Min. Zentraleinheitszeit benutzen und muß 2 Std. und 50 Min. nach der Startzeit erledigt sein.

7: Es dürfen max. 150 Karten gestanzt werden.

8: Die Startzeit des Jobs ist 9.30 Uhr.

Es gibt bei dem Betriebssystem EXEC 8 wesentlich mehr Executive-Control-Language-Statements als beim Operating System der IBM. Die verschiedenen Statements erlauben dabei entweder eine Verschlüsselung in einer Form, die dem System bei der Zuordnung der einzelnen Systemkapazitäten viel Spielraum läßt oder sie geben eine sehr genaue Festlegung für das Vorgehen des Systems vor. In dem Bereich, der beim OS als Data Management bezeichnet wurde, bietet das EXEC 8 die gleichen Möglichkeiten, Datenbestände zu katalogisieren, verschiedene Generationen aufzubauen und diese mit einer absoluten oder relativen Verschlüsselung aufzurufen wie das Operating-System.

Das eigentliche Einplanungsverfahren weicht allerdings in starkem Maße von dem des Operating-Systems ab. Der Hauptbestandteil des EXEC 8 ist der sog. Supervisor [1], der aus einer Reihe von Routinen besteht, die teilweise kernspeicherresi-

1) Der Supervisor kontrolliert die Ausführung aller Jobs. Zu seinen Routinen gehören u.a. die Einplanungs-, Unterbrechungs-, Zeiteinteilungs- und Logbuch-Routinen.

dent sind, teilweise aber auch nur bei Bedarf in den Kernspeicher geladen werden. Das EXEC 8 kennt wie die MVT-Version des OS keine fest definierten Kernspeicherbereiche, sondern ordnet den Kernspeicher dynamisch zu. Im Gegensatz zum MVT ermöglicht es aber auch eine Re-Allokierung des Kernspeichers, d.h. es kann die Programme so zusammenschieben, daß nach der Beendigung einer Task wieder zusammenhängende freie Kernspeicherbereiche entstehen. Dieses Zusammenschieben wird aber nach Möglichkeit vermieden, da es sehr zeitaufwendig ist.

Für die Auswahl der Jobs, die initialisiert werden sollen, ist der sog. Coarse Scheduler [1] zuständig. Die Auswahl erfolgt dabei zunächst in der Gruppe mit der höchsten Priorität. Initialisiert wird der Job, dessen Kapazitätsanforderungen hinsichtlich der peripheren Geräte erfüllt werden können. Das EXEC 8 ordnet zunächst die Peripherie zu und gibt dann den Job weiter zur Zuweisung von Kernspeicher und zur Ausführung. Der Coarse Scheduler bedient sich bei seiner Arbeit folgender Informationen:

1. der Priorität,
2. der Deadline-Zeit,
3. der geschätzten Laufzeit und
4. der Startzeit.

Eine Besonderheit des Systems besteht bei der Einplanung der Deadline-Läufe. Sobald die Ausführung der Läufe bis zu dem vorgegebenen Zeitpunkt oder während der zur Verfügung stehenden Zeitspanne gefährdet erscheint, erhöht der Coarse Scheduler intern die Priorität dieses Jobs, um eine fristgemäße Beendigung sicherzustellen.

Die Kernspeicherzuweisung und die Ausführung der Job-Steps werden vom Dynamic Allocator [2] überwacht, der die Steps ihrem Abgabetermin entsprechend in der nachfolgenden Reihenfolge abarbeitet:

---

1) Der Coarse Scheduler ist mit den Initiatoren des OS vergleichbar. Er interpretiert die Steuerkarten der eingelesenen Jobs und ordnet sie ihrer Priorität entsprechend in Warteschlangen ein.

2) Der Dynamic Allocator ist für die Vergabe der CPU-Zeit sowie des Kernspeichers an die einzelnen Jobs verantwortlich.

1. "critical deadline"-Steps, für die auf Grund der Deadline-Angabe unverzüglich Kernspeicher bereitgestellt werden muß, und zwar ohne Rücksicht auf andere Tasks, die evtl. zurückgestellt und auf einen Plattenspeicher ausgelagert werden. Critical deadline-Läufe besitzen die höchste Priorität.
2. "semi-critical deadline"-Steps, für die ebenfalls Kernspeicher bereitgestellt werden muß. Es erfolgt hier aber keine Unterbrechung laufender Tasks, sondern die Kernspeicherzuordnung wird solange gestoppt, bis die Anforderungen des Jobs erfüllt werden können.
3. "deadline"-Steps: Der Dynamic Allocator besitzt hier eine Wahlmöglichkeit zwischen verschiedenen Deadline-Läufen. Er wählt den aus, der den freien Kernspeicherbereich bestmöglich ausnutzt.
4. "normal"-Steps: Auch hier wird der Job-Step ausgewählt, der den Kernspeicher am besten ausnutzt. Bei gleicher Kernspeicheranforderung wird dem Step mit der größeren Priorität Vorrang verschafft.

Es fällt bei der Arbeitsweise des Dynamic Allocator auf, daß man hier vom reinen Prioritätsprinzip abweicht und immer die Task wählt, die den freien Kernspeicherbereich am besten ausfüllt. Auf das genaue Vorgehen des Dynamic Allocator soll nicht näher eingegangen werden. Erwähnt werden muß noch, daß sowohl der Coarse Scheduler als auch der Dynamic Allocator die Tasks von Zeit zu Zeit hinsichtlich der zur Verfügung gestellten und der auf Grund der geschätzten Zeitangabe noch zur Verfügung zu stellenden Zentraleinheitszeit untersuchen und nach einem bestimmten Schema gewichten, um immer die Task mit der höchsten Dringlichkeit zu erkennen. Die grundlegende Zielrichtung in der Arbeit des Dynamic Allocator ist also in dem Versuch zu sehen, eine termingerechte Erledigung der Batch-Läufe sicherzustellen.

Die Zuteilung der Zentraleinheit übernimmt der sog. Dispatcher, der eine vom Dynamic Allocator nach Prioritätsgesichtspunkten aufgebaute Tabelle (Switch-Liste) abarbeitet, die von diesem ständig überwacht und korrigiert wird.

Der Algorithmus, nach dem der Dispatcher die CPU vergibt, hat folgenden Aufbau:

1. Programme des Levels L besitzen stets Priorität gegenüber Programmen des Levels L+1. (Es handelt sich bei dieser Einteilung um intern errechnete Prioritäten, die bei Batch-Programmen zunächst mit den extern eingegebenen Prioritäten übereinstimmen.)
2. Programme eines Levels werden gleichwertig behandelt.
3. Ein Programm verliert die Kontrolle über die Zentraleinheit durch freiwillige Freigabe (z.B. E-/A-Operation) oder durch Überschreiten eines Zeitlimits Q[1], das vom Dynamic Allocator nach der Beziehung $Q = A \cdot (1 + P/F) \cdot T$ festgesetzt wird. Dabei ist A ein Zuordnungsfaktor, der vom Dynamic Allocator vergeben wird, P die Prioritätsziffer des Jobs aus dem RUN-Statement, F ein bei der Systemgenerierung zu definierender zusätzlicher Prioritätsfaktor und T ein Zeitfaktor, der in Zusammenhang mit dem Level des Jobs steht ($T = 2^L$)[2]. Überschreitet eine Task das so festliegende Limit, wird sie auf den nächst größeren Level verschoben, d.h. ihre Priorität sinkt, aber ihr Zeitlimit steigt für die nächste CPU-Vergabe an.

Das EXEC 8 erkennt also selbsttätig, ob ein Programm rechen- oder E-/A-intensiv ist und ändert auf Grund des Verhaltens der Programme dynamisch die Beziehungen zwischen Prioritätsstufe und Größe der Zeitscheibe, um zu einer optimalen Ausnutzung der Zentraleinheit zu kommen. Die vom Benutzer in der RUN-Karte definierte Priorität dient nur der Vergabe der externen Peripherie und dem erstmaligen Einordnen eines Jobs in eine Warteschlange, die der Coarse Scheduler abzuarbeiten hat. Die weitere Behandlung des Runs wird vom System selbsttätig und automatisch abgewickelt. Auch hier wird deutlich, daß der Anwender sich darauf zu beschränken hat, die Jobs so anzuordnen, daß die Peripherie möglichst weitgehend ausgenutzt wird. Gleichzeitig muß dabei geprüft werden, ob der zur Verfügung

---

1) Die Zeiteinheit für dieses Zeitlimit geht aus den Unterlagen, die dem Verfasser zur Verfügung standen, nicht hervor. Sie wird wahrscheinlich mit der Zykluszeit der Zentraleinheit übereinstimmen.

2) Die Zuordnung $T = 2^L$ soll sicherstellen, daß der Zeitfaktor T mit wachsendem Level L (also sinkender Priorität eines Jobs) zunimmt.

stehende Kernspeicher ausreicht. Auf die Zuteilung des Kernspeichers, der Zentraleinheit und der Ein-/Ausgabekanäle während der Programmausführung besitzt er keine Einflußmöglichkeit.

2.5 SIEMENS: Platte-Betriebssystem (PBS) [1)2)]

Das Platte-Betriebssystem wurde für das Siemens-System 4004 entwickelt. Es umfaßt ca. 50 aufeinander abgestimmte Systemprogramme, die sich wiederum in die drei Gruppen

1. Organisationsprogramm,
2. Übersetzer und
3. Dienstprogramme

aufteilen lassen. Wie in den vorhergehenden Abschnitten soll auch beim PBS nur das Organisationsprogramm dargestellt werden. Außerdem soll in diesem Abschnitt das im Juli 1971 vorgestellte Programm "PBS 4004 - Dynamische Anlagenbelegung DAB 3" [3)] erläutert werden, das einige für diese Arbeit sehr wichtige Probleme aufwirft.

Das Organisationsprogramm gliedert sich in den Ablaufteil, das Ein-/Ausgabesystem, den Monitor und das Datenübertragungssystem.
Der Ablaufteil ist das zentrale Programm innerhalb des Organisationsprogramms und enthält

1. die Routine zur Unterbrechungsanalyse,
2. die Ein-/Ausgabesteuerung,
3. die Fehlerbehandlungsroutine,
4. die Programmsteuerung und
5. den Bedienungsteil für die Konsolschreibmaschine.

Zur Erledigung seiner Steuerungsfunktionen bedient er sich bestimmter Ta-

---

1) SIEMENS System 4004, 4004/35, 135, 45, 55, 60, 150 Platte-Betriebssystem, Beschreibung, Best.-Nr. D 14/4014.

2) SIEMENS, PBS, Organisationsprogramm, München 1970.

3) PBS 4004 - Dynamische Anlagenbelegung DAB, Best.-Nr. D 14/4144, München 1971.

bellen, auf Grund derer ständig überprüft wird, ob noch zusätzliche Programme ausgeführt werden können.

Die Ablaufprioritäten der einzelnen Jobs werden durch sog. Befehlszähler festgelegt, die der Ablaufteil beim Laden der Programme vergibt. Maximal können 14 Befehlszähler vergeben werden, d.h. es können 14 Jobs simultan ablaufen. Die Programm-Ablaufprioritäten wachsen mit den Befehlszählernummern (1,2,3,...,9,A,B,C,D,E,); der dem Befehlszähler E zugeordnete Job hat also im Normalfall die höchste Ablaufpriorität [1]. Die Zuordnung des Befehlszählers kann dabei entweder im Ladeaufruf durch den Operator vorgenommen oder dem Ablaufteil überlassen werden. Der Ablaufteil wählt entweder den freien Befehlszähler niedrigster Priorität oder den freien Befehlszähler höchster Priorität, je nachdem wie die Festlegung bei der Systemgenerierung erfolgte. Wichtig ist, daß immer nur freie Befehlszähler vergeben werden können, d.h. die einzelnen Programme besitzen von Lauf zu Lauf eine unterschiedliche Ablaufpriorität [2]. Ein belegter Befehlszähler wird mit dem Ende des Programms wieder freigegeben.

Der Kernspeicher wird dem Programm beim Ladeaufruf zugeordnet. Die Länge und Lage des benötigten Kernspeichers kann entweder im Ladeaufruf angegeben oder die Zuteilung kann dem Ablaufteil überlassen werden. Dabei ist darauf zu achten, daß der zur Verfügung gestellte Kernspeicherbereich den Anforderungen des größten Steps genügen muß. Eine Neuordnung des Kernspeichers nach Beendigung eines Steps kennt das PBS nicht; es werden also keine zusammenhängenden freien Kernspeicherbereiche geschaffen.

Die Zuordnung der Ein-/Ausgabeeinheiten kann an einen Befehlszähler gebunden werden, es kann aber auch bestimmt werden, daß eine Zuordnung für mehrere Befehlszähler gilt.

---

1) Die Prioritäten können vom Operator auch in anderer Reihenfolge eingegeben werden.

2) Dieses Vorgehen entspricht in etwa der MFT-Version des IBM-Operatingsystems.

Um gegenseitige Störungen im Multiprogramming-Betrieb zu vermeiden, gibt es die Möglichkeit, Einheiten für die Benutzung durch andere Programme zu sperren, so daß nur ein Programm mit der Einheit arbeiten kann. Diese Zuordnung der Ein-/Ausgabeeinheiten im PBS ist nicht unproblematisch. Die Einheiten müssen in den Steuerkarten immer mit ihren symbolischen Gerätenamen aufgerufen werden, und das bedeutet für den Programmierer, daß er sich schon im voraus für ganz bestimmte Geräte entscheiden muß. Benötigt er z.B. drei Magnetbandstationen und stehen 7 gleichwertige Einheiten bei der Anlage zur Verfügung, so muß er sich in den Steuerkarten für drei ganz bestimmte Bandeinheiten entscheiden, denn das Betriebssystem besitzt nicht die Möglichkeit, ihm im Zeitpunkt der Gerätebelegung drei beliebige, freie Einheiten zuzuordnen. Deshalb kann es im Multiprogramming-Betrieb sehr leicht zu Engpässen kommen, weil mehrere Programme die gleichen Einheiten angefordert haben, obwohl andere, gleichwertige Geräte nicht belegt sind. Diese Art der Zuordnung stellt an den Konsol-Operator bzw. an die Arbeitsvorbereitung hohe Anforderungen[1].

Der Monitor ermöglicht den automatischen Ablauf einer Folge von unabhängigen Programmen, deren Steuerinformationen in einer Bibliothek gespeichert werden und dient so der Entlastung des Operators. Maximal können 14 Monitorläufe parallel ablaufen. Die einzelnen Programme, die dabei unter einem Monitor ablaufen, besitzen alle den gleichen Befehlszähler und damit die gleiche Ablaufpriorität. Belegte Geräte können im Rahmen eines Monitorlaufs von einem Programm an das folgende weitergegeben werden. Die Freigabe erfolgt erst mit dem Ende des Monitorlaufs. Die Verbindung mehrerer Programme zu Monitorläufen entspricht beim OS der IBM die Möglichkeit, mehrere aufeinanderfolgende Steps zu einem Job zusammenzufassen.

Auf das Ein-/Ausgabesystem sowie das Datenübertragungssystem soll nicht eingegangen werden, da diese Systemroutinen für eine Belegungsplanung von Großrechensystemen nebensächlich sind.

---

1) Eine gewisse Erleichterung bietet das Hilfsprogramm DAB-3, auf das noch näher eingegangen wird.

Das Platte-Betriebssystem bietet von den vorgestellten Betriebssystemen zweifellos den geringsten Komfort, und zwar aus folgenden Gründen:

1. Die Einheitenzuordnung ist für ein Multiprogramming Betriebssystem unzureichend und stellt an die Arbeitsvorbereitung und den Konsoloperator so hohe Anforderungen, daß bei einem Simultanablauf von mehr als fünf Programmen (Monitoren) eine optimale Ausnutzung der Anlage kaum mehr möglich ist.
2. Das Organisationsprogramm enthält keine Routinen für das sog. "Data Management" und erschwert dadurch die Dateipflege und -verwaltung in starkem Maße, da die Dateien nur durch ihre Rollen-Nummern bzw. ihre Lage auf Plattenspeichern identifizierbar sind. Die Fehleranfälligkeit eines solchen Systems ist deshalb als sehr groß anzusehen.

Mit dem Programm Dynamische Anlagenbelegung DAB 3 bietet SIEMENS seit Mitte 1971 eine Ergänzung zum PBS an, die eine wesentliche Verbesserung des Betriebssystems darstellt. Da das in diesem Programm beschriebene Verfahren für die vorliegende Arbeit von Bedeutung ist, soll es ausführlicher besprochen werden.

Das Programm DAB 3 versucht, den Multiprogrammbetrieb maschinell durch eine dynamische Anlagenbelegung zu steuern[1)]. Es ist dementsprechend notwendig, DAB 3 ständig im Kernspeicher zur Verfügung zu haben. Es wird alle 20 Sekunden aktiv und prüft, ob ein geladenes Programm beendet wurde. Ist dies der Fall, wird unter den zur Verfügung stehenden Programmen, den Einplanungskriterien entsprechend, das günstigste ausgesucht und als Ladevorschlag ausgedruckt. Ein neuer Vorschlag wird erst ausgedruckt, wenn der Operator den ersten Vorschlag geladen oder dem DAB 3 mitgeteilt hat, daß das entsprechende Programm übergangen werden soll. Ein übergangenes Programm wird immer als beendet angesehen, seine Parameterkarte[2)] muß bei Bedarf neu eingegeben werden. Neben der Auftragsnummer werden im

---

1) s. S. 5.

2) Die Parameterkarten enthalten die Kapazitätsanforderungen der Monitorläufe.

Ladevorschlag von DAB 3 die vorgeschlagene Befehlszähler-Nummer, der Kernspeicherbedarf sowie die neu einzugebende Prioritätenfolge genannt. Der Operator hat zusätzlich die Möglichkeit, dem DAB 3 nicht bekannte Programme unter der angegebenen Befehlszähler-Nummer auszuführen; er ist also nicht gezwungen, sich an den Vorschlag zu halten.

Für die Optimierung des Programmdurchlaufs stehen dem Benutzer in DAB 3 drei Kriterien zur Verfügung:

1. Die Gewichtsoptimierung , bei der an Hand der eingelesenen Parameterkarten das Programm ausgewählt wird, das eine bestmögliche Kernspeicherausnutzung bei bestmöglicher Peripherie-Auslastung und dadurch die kürzeste Anlagenbelegungsdauer für die Gesamtheit der Aufträge gewährleistet.
2. Eine Kurzläuferoptimierung , bei der immer das Programm mit der kürzesten Laufzeit ausgewählt wird, und zwar ohne Rücksicht auf die Auslastung der Anlage.
3. Die Langläuferoptimierung , bei der das Programm mit der größten Laufzeit vorgeschlagen wird, ebenfalls ohne Beachtung des Auslastungsgrades der Anlage.

Um die Programme[1] nach einem dieser Kriterien einplanen zu können, wertet DAB 3 verschiedene Tabellen des Ablaufteils aus, ohne diesen aber direkt zu beeinflussen. Das Programm kann in der Standardgröße bis zu 40 Parameterkarten speichern, die den in Tab. 5 dargestellten Aufbau besitzen[2].

---

1) Die Programme bzw. Programmfolgen müssen für DAB 3 immer als Monitorläufe eingegeben werden.

2) PBS - Dynamische Anlagenbelegung - DAB 3, a.a.O., S. 13.

Tab. 5: Übersicht über die Angaben in der Parameterkarte für das Programm DAB 3

Spalte [1]	Bezeichnung
*) 11 - 14	Rechenzeitbedarf in Minuten
24 - 31	Vorprogramm
*) 32 - 39	Auftragskennung
*) 43 - 45	Kernspeicherbedarf in KB
46 - 58	Anzahl benötigter Geräte
46	Lochstreifenlesestanzer
47	Magnetbandgeräte 7-Spur
49	Lochkartenstanzer
51 - 52	Magnetbandgeräte 9-Spur
53 - 54	Plattenspeicher 564
56	Groß-Plattenspeicher 590
58	Y-Gerät (z.B. Plotter)
*) 59 - 60	Intensität der Zentraleinheit-Belegung in Prozent zur Verweilzeit
61 - 62	Geräteart für Y-Gerät, z.B. OE für Trommelspeicher, 08 für Belegleser
Die mit *) bezeichneten Eintragungen müssen vorhanden sein.	

Das Programm DAB 3 benötigt mindestens 5688 Bytes. Ein neuer Vorschlag wird in ungefähr einer Sekunde ermittelt. Wird kein neuer Vorschlag gemacht, dauert die Unterbrechung ca. 10 Millisekunden. Damit löst DAB 3 das Problem der Laufzeitschätzung für Programme, die im Multiprogramming-Betrieb verarbeitet werden, zweifellos auf eine sehr elegante Art. Bedingt durch die sehr geringe Programmgröße werden aber viele, in Großrechenzentren ständig wiederkehrende Probleme nicht gelöst. Besonders in den folgenden Punkten ist das Programm noch verbesserungsbedürftig:

1. Ein Optimum hinsichtlich der Anlagenbelegung kann von DAB 3 nur zufällig gefunden werden, denn es wird nicht versucht, die für die Gesamtheit der auszuführenden Programme benötigte minimale Verarbeitungszeit zu bestimmen, sondern es wird immer nur (bei dem 1. Optimierungskriterium) dasjenige Programm gewählt, das die

1) Die nicht angegebenen Spalten bleiben frei.

Anlage für einen gegebenen Zeitpunkt optimal auslastet. Dieses auf einen Zeitpunkt bezogene Optimum muß aber keineswegs mit dem Gesamtoptimum übereinstimmen.

2. Eines der Hauptprobleme von Großrechenzentren, nämlich die automatische Feststellung der Abhängigkeiten zwischen einzelnen Programmen, bleibt der Arbeitsvorbereitung überlassen[1]. Dieses Problem kann von DAB 3 wegen der geringen Programmgröße auch kaum gelöst werden.
3. Der Planungsalgorithmus des DAB 3 liegt fest. Bei der Gewichtsoptimierung soll eine bestmögliche Kernspeicherbelegung bei bestmöglicher Peripherie-Auslastung erfolgen. Es scheint aber sinnvoller zu sein, zunächst einmal festzustellen, welche peripheren Geräte bei einer gegebenen Anlagenkonfiguration Engpaßfaktoren sind, um die Belegung an diesen Faktoren orientieren zu können.
4. Die vorgeschlagene Kurzläuferoptimierung ist ein sehr schlechtes Optimierungskriterium, da die sehr lange laufenden Programme am Ende eines Planungszeitraums voraussichtlich zu einem sehr schlechten Auslastungsgrad führen.
5. Es kann nur ein Vorprogramm angegeben werden. In Rechenzentren zeigt sich hingegen sehr häufig, daß ein Programm weit mehr als nur ein Vorprogramm besitzt.
6. Viele Programme dürfen nicht parallel zueinander ablaufen, weil sie z.B. gleiche Stammdateien benötigen. Dieses Problem ist in DAB 3 nicht gelöst.

Obwohl diese Punkte an der Einsatzfähigkeit des Programms DAB 3 in Großrechenzentren zweifeln lassen, ist es als Versuch einer Herstellerfirma, dem Benutzer einer Datenverarbeitungsanlage ein Software-Programm mitzuliefern, das einen möglichst guten Auslastungsgrad ermöglichen soll, hoch zu werten.

---

1) Einschränkend muß hier allerdings gesagt werden, daß sich das PBS auf Grund des fehlenden Data Managements nur sehr schlecht zu einer maschinellen Feststellung von Programmstrukturen eignet.

3 AUFBAU EINER JOBSTAMMDATEI

3.1 Die Aufgaben einer Jobstammdatei

In einem Großrechenzentrum gibt es durchschnittlich insgesamt 1 000 - 2 000 Routinejobs[1], von denen pro Tag - in unterschiedlichen Zeitabständen - 150 bis 200 Jobs verarbeitet werden. Für den einzelnen Sachbearbeiter bzw. Arbeitsvorbereiter ist es bei dieser Größenordnung praktisch nicht mehr möglich, einen Überblick über die logischen Zusammenhänge der Jobs untereinander und auch über einzelne Jobs zu behalten. Selbst bei kleineren Arbeitsgebieten treten Schwierigkeiten auf, sobald auch andere Arbeitsgebiete über Dateien bzw. Dateiveränderungen Einfluß auf dieses Gebiet nehmen. In jedem Rechenzentrum stellt sich daher die Aufgabe, ein Instrument aufzubauen und ein Verfahren zu entwickeln, das der Arbeitsvorbereitung in kürzester Zeit einen genauen Überblick über die in einer Planungsperiode[2] zu erledigenden Routineprogramme verschaffen kann. Diese Forderung stellt sich vor allem, weil die Hersteller von EDV-Anlagen bei ihren Multiprogramming-Betriebssystemen davon ausgehen, daß die einzelnen Jobs, die parallel ablaufen, unabhängig voneinander sind. Je stärker sich aber die in letzter Zeit festzustellende Integration großer Arbeitsgebiete in der Datenverarbeitung durchsetzt, desto schwieriger wird es, für den Arbeitsablauf voneinander unabhängige Jobs zusammenzustellen. Es läßt sich zeigen, daß der größte Teil der Routinejobs an logische Abhängigkeiten gebunden ist. So ist z.B. auf den Gebieten der Auftragsbestandsführung, der Materialbestands-

---

1) Unter Routinejobs versteht man Programme bzw. Programmfolgen, die in regelmäßigen oder unregelmäßigen Zeitabständen ablaufen und die das Teststadium verlassen haben.

2) Als Planungsperiode wird in der Regel der Arbeitstag angesehen. Andere Planungszeiträume (z.B. Schichten) sind aber genauso denkbar.

führung und der Auftragsabrechnung in den letzten zwei bis drei Jahren ein deutlicher Trend zu integrierten Bestandsführungssystemen zu erkennen, der zu einer starken Verflechtung der einzelnen Jobs geführt hat.

Als ein geeignetes Instrument zur Klärung der Fragen, die im Zusammenhang mit Routinearbeiten auftreten können, ist eine sog. Jobstammdatei anzusehen. Sie enthält für jeden Routinejob jeweils einen Satz, der die relevanten Daten des Jobs - seine Stammdaten - beschreibt. Der Satzaufbau und die Satzlänge werden naturgemäß von Benutzer zu Benutzer variieren. Es hat sich aber gezeigt, daß man in einem ca. 900-stelligen Satz die wichtigsten Jobstammdaten speichern kann. Als Speichermedium kann sowohl ein Magnetband als auch eine Magnetplatte Verwendung finden.

Die Jobstammdatei kann vor allem drei Zielen dienen:

1. Durchführung einer automatisierten täglichen Ablauf- und Belegungsplanung (Tagesplanung)[1]. Auf Grund der in jedem Jobstammsatz gespeicherten Ein- und Ausgabedateien besteht die Möglichkeit, nahezu alle logischen Abhängigkeiten der Jobs untereinander sichtbar zu machen, und durch das im nächsten Abschnitt noch näher zu erläuternde Terminsteuerwort können mit Hilfe eines einfachen Programms alle Jobs aus der Jobstammdatei ausgewählt werden, die an einem bestimmten Planungstag laufen müssen. Die Aufstellung des sog. Tagesplans, mit dem sonst die Sachbearbeiter der einzelnen Arbeitsgebiete täglich mindestens 15 - 45 Minuten belastet waren, kann dann mit einem Programm in ca. 10 Minuten (bei ungefähr 150 - 200 Jobs pro Tag) erledigt werden.

2. Durchführung einer monatlichen Vorausschau für den Auslastungsgrad der Anlage (Monatsplanung). Für die Leitung eines Groß-

---

1) Die Ablaufplanung wird im einzelnen in Kapitel 4, die Belegungsplanung in Kapitel 6 dargestellt.

rechenzentrums ist die Vorausschätzung der Anlagenbelastung für einen Zeitraum von ca. 4 Wochen von größter Bedeutung. Nur wenn Belegungsspitzen frühzeitig genug erkannt werden, ist es möglich, durch Terminverschiebungen bei voneinander abhängigen Jobs zu einer gleichmäßigen Auslastung zu kommen. Die Jobstammdatei ermöglicht über eine Auswertung der Terminsteuerworte sowie der Verarbeitungs- und Druckzeiten die Aufstellung eines Monatsplans, auf Grund dessen ein gezielter Abbau von Arbeitsspitzen bei Routinearbeiten und die Entwicklung eines Zeitplans für notwendige Testläufe möglich ist.

3. Benutzung der Jobstammdatei als Informationssystem für rechenzentrumsinterne Belange und für externe Benutzer.

Der Bedarf eines Rechenzentrums an Auswertungen der Jobstammdatei über die bestehenden Routinejobs ist recht groß. Als besonders interessant haben sich die folgenden Auswertungen erwiesen:

1. Kenntlichmachung der Jobstrukturen für verschiedene Arbeitsgebiete und evtl. Konzerngesellschaften unter Beachtung der unterschiedlichen Laufhäufigkeiten;
2. allgemeine Aufstellung über die Laufhäufigkeiten von Jobs;
3. Auswertungen über Bandeinheiten- und Plattenspeicherbelegung sowie über Lade-, Verarbeitungs- und Druckzeiten;
4. Überprüfung der Terminsteuerworte und
5. Feststellung von Cross-Referenzen zwischen Jobs, Steps, Dateien und Sätzen.

Damit die Datei eine aussagefähige Basis für solche Auswertungen darstellt, ist sie stets auf dem neuesten Stand zu halten, d.h. Neuzugänge, Änderungen und Abgänge von Jobs sind in der Jobstammdatei unverzüglich festzuhalten.

3.2 Die Definition eines Jobstammsatzes

Die Definition eines einzelnen Jobstammsatzes ist vor allem von den Zielen abhängig, die ein Anwender mit der Jobstammdatei erreichen will. Benutzt er sie hauptsächlich als Informationssystem, muß der Grad der Untergliederung bei den einzelnen Daten wesentlich größer sein als wenn er die Datei nur für Planungszwecke einsetzt. Der nachfolgend dargestellte Satzaufbau ist primär an der Aufgabenstellung dieser Arbeit orientiert, eine Belegungsplanung für einen Großrechner durchzuführen. Aus diesem Grund erfolgte auch eine Beschränkung auf die reinen Jobdaten, und die Einzelinformationen aus den Steps blieben unberücksichtigt, obwohl sie für statistische Auswertungen häufig von großem Interesse wären. Die Satzlänge der einzelnen Sätze ist aus Gründen der Speicherungstechnik als fest angenommen, man kann aber genausogut auch mit variabler Satzlänge oder mit einem festen und einem variablen Satzteil arbeiten. Vom Verfasser wurde in FORTRAN IV ein Programm für den Aufbau und den Änderungsdienst einer Jobstammdatei geschrieben, die zur Zeit die Stammdaten von etwa 1 800 Routinejobs enthält[1].

Als Eingabe für den Aufbau der Datei dienen Lochkarten, die teils maschinell aus den Job-Control-Steuerkarten des OS gewonnen und teils manuell erstellt werden. Eine Aufgliederung der Stammdaten auf sieben verschiedene Datentypen hat sich dabei vor allem für den Änderungsdienst als zweckmäßig erwiesen. Ein Jobstammsatz läßt sich in folgende Informationsgruppen gliedern:

1. Jobbeschreibungsdaten,
2. Zeiterfordernisse des Jobs,
3. Maschinenerfordernisse,
4. Druckerfordernisse,

---

1) Der eigentliche Aufbau dieser Datei ist von einem Arbeitsvorbereiter in etwa zwei Monaten vollzogen worden.

5. Kostenumlageinformationen (in %),
6. Folgejobangaben und
7. Fileinformationen.

Die vom Verfasser aufgebaute Jobstammdatei ist sequentiell - nach Jobnamen sortiert - auf einem Magnetplattenspeicher gespeichert. Das Programm gibt zu jedem Jobstammsatz ein sog. Jobstammblatt aus, das der Arbeitsvorbereitung und den Sachbearbeitern bei Einzelfragen als Informationsgrundlage dient. An Hand des Jobstammblattes, das Abb. 7 zeigt, sollen einzelne Stammdaten, den Informationsgruppen entsprechend, näher erläutert werden[1]:

zu 1. Jobbeschreibungsdaten

Jobname (1)

Der Jobname dient der Identifikation des Jobs. Er ist gleichzeitig Sortierkriterium für die Jobstammdatei. Im Beispiel handelt es sich um einen 7-stelligen alphanumerischen Jobnamen. In ihm kann außerdem verschlüsselt werden, ob es sich um einen bereits laufenden Routinejob handelt oder um einen Job, der zunächst nur geplant ist. Die Aufnahme geplanter Jobs in die Jobstammdatei wird notwendig, wenn man mit ihrer Hilfe eine in Kapitel 8 näher beschriebene mittelfristige Systemkonfigurierung durchführen will. Eine Aufnahme von Test- bzw. Probeläufen in die Jobstammdatei empfiehlt sich nicht, da sie einen zu kurzfristigen Charakter besitzen und den Umfang der Datei unnötig erweitern würden.

Maschinen-Nummer (2) : 65

Die Angabe der Maschinen-Nummer erübrigt sich, wenn das Rechenzentrum nur über eine Anlage verfügt; andernfalls dient sie sowohl für Planungs- als auch für allgemeine Auswertungszwecke als Auswahlkriterium. Hier dient sie als Abkürzung für eine Anlage des Typs IBM /370-165.

---

1) Die Ziffern hinter den Jobstammdaten beziehen sich jeweils auf die in Kreise gesetzten Ziffern des Jobstammblattes aus Abb. 6.

Abb. 7: Beispiel eines Jobstammblattes für Job J 999 999

```
                                  J O B S T A M M B L A T T      V O M     7.12.1972
                    (1)       (2)  (3)  (4)                (5)
J O B N A M E     J999999     65   004  711      HRW STW TAGESBERICHT

1. ZEITERFORDERNISSE UND JOB-DEFINITION
                 (6)                          (7)   (8)        (9)    (10)  (11)   (12)
        TERMINSTEUERWORT                PREIS  SA   BETE     ABGABE   LE    VZ   DZWP  DZDB
 1 1                 25                 1240   01   0900      1600    2    210    15    10

2. MASCHINENERFORDERNISSE
   (13)      (14)      (15)      (16)    (17)          (18)
   ANZ.      PRTY   REGION   K.SP.   INT.  CLASS  RANDOM   RANDOM   RANDOM   RANDOM   RANDOM   RANDOM
   STEPS            BED.                          SP.KL.   SP.GR.   SP.KL.   SP.GR.   SP.KL.   SP.GR.
    28       08      104      104     30      C       B      0264

3. DRUCKERFORDERNISSE
                 (19)                  (20)
    ANZ.    DRUCKAUSGABE-   ANZ.      ANZ.
   DR.KL.      KLASSEN      DRUBA     LIST.
      3         BJ             2         6

4. FIRMENEINLAGE : FIRMA - PROZENT                (21)
   002 60  004 20  007 20

5. FOLGEJOBS      J900005     J900020             (22)

6. EIN-/AUSGABEFILES                              (23)

   EING.FIL  GEN  DISP   EING.FIL  GEN  DISP   EING.FIL  GEN  DISP      AUSG.FIL  GEN   AUSG.FIL  GEN
   A.F10100   0     0    C.R30034         S    I.F90323   0    0        A.F10100  +1    A.F90426  +1
```

Firmen-Nummer (3)

Bei Konzerngesellschaften können hier einzelne Tochterfirmen verschlüsselt werden, bei Einzelunternehmen kann die Abteilung benannt werden, für die der Job läuft. Diese Angabe ist vor allem für Auswertungszwecke wichtig, sei es, daß man Jobstrukturen für eine Tochtergesellschaft aufzeigen will, sei es, daß irgendwelche anderen Auswertungen von der Tochtergesellschaft verlangt werden.

Programmierer-Nummer (4)

Es empfiehlt sich, die Programmierer-Nummer zu speichern, um a) bei einem Jobabbruch sofort zu wissen, wer zu benachrichtigen ist und b) bei Jobänderungen eines Arbeitsgebiets, das auch in andere Gebiete hereinreicht, schnell zu erfahren, wer welche Programme zu ändern hat.

Kurzbeschreibung des Jobs

Hier kann eine 45-stellige Kurzbeschreibung des Jobs gespeichert werden, die in gedrängter Form die Problemstellung des Jobs umschreibt.

zu 2. Zeiterfordernisse des Jobs

Terminsteuerwort (6)

Dem Terminsteuerwort ist innerhalb der Jobstammdatei für Planungszwecke eine entscheidende Bedeutung beizumessen, denn in ihm wird verschlüsselt, an welchen Tagen im Monat ein Job läuft. Es ist die Aufgabe der Sachbearbeitergruppen oder besser noch einer besonderen Termingruppe in monatlichen Abständen in Abstimmung mit den Fachabteilungen Lauftermine für die einzelnen Routinejobs festzusetzen und darauf zu achten, daß diese Termine eingehalten bzw. Änderungen unverzüglich neu in die Jobstammdatei eingegeben werden. Aus Speicherungs- und Auswertungsgründen sollte man sich bemühen, das Terminsteuerwort so kurz wie möglich zu halten, selbst wenn dadurch nicht alle möglichen Lauftermine und Laufhäufigkeiten verschlüsselt werden können. Da meist sowieso nicht sämtliche Jobs terminlich zu fixieren sind und dann als sog. Abruf-

jobs in die Planung einbezogen werden müssen, kann man diese Möglichkeit auch bei Jobs ausnutzen, die in sehr unregelmäßigen Zeitintervallen laufen.

In den meisten Fällen kommt man mit einem Steuerwort aus, das beispielsweise folgenden Aufbau besitzt:

Abb. 8: Aufbau des Terminsteuerworts

| Spalten | | |

e) monatlich einmal laufende Jobs: Lochung in den Spalten 1 - 7 und 8 - 11
oder
Lochung in den Spalten 25, 26 oder
Lochung in den Spalten 27, 28 oder
Lochung in der Spalte 29 oder
Lochung in der Spalte 30
Jobs, die jeden Monat an einem anderen, festlegbaren Tag laufen, müssen monatlich von der Sachbearbeitung neu aufgegeben werden;

f) Jobs, die in viertel-, halbjährlichen oder jährlichen Intervallen laufen, sind wie monatlich laufende Jobs zu verschlüsseln und erhalten zusätzlich noch Lochungen in den entsprechenden Monatsspalten 13 - 24;

g) Laufen Jobs monatlich zweimal, kann zusätzlich zu den Lochungen in den Spalten 25, 26 oder 27, 28 noch in den Spalten 31, 32 eine Zahl mit "xx" angegeben werden. Der Job erscheint dann nach xx Tagen automatisch wieder im Tages- bzw. Monatsplan.

Der Job J 999999 läuft beispielsweise jeweils dienstags und donnerstags sowie außerdem am 25. Kalendertag jeden Monats.

Durch diese Art der Verschlüsselung der Lauftermine lassen sich natürlich nicht alle Möglichkeiten erfassen[1]. Es besteht dann aber jederzeit die Möglichkeit, diese Jobs als Abrufjobs zu kennzeichnen, die nur nach Anforderung durch die Sachbearbeitung bzw. Termingruppe laufen. Die Zahl solcher Abrufjobs sollte in einem Rechenzentrum jedoch so klein wie möglich gehalten werden, um die Aussagefähigkeit des Monatsplans nicht zu sehr zu beeinflussen.

Genügt das vorgeschlagene Terminsteuerwort den Anforderungen eines Rechenzentrums nicht, lassen sich ohne große Schwierigkeiten andere Steuerworte entwickeln, die dann aber meist speicheraufwendiger sein werden.

Sachbearbeitergruppe (7)

Vor allem in Großrechenzentren werden für die Eingabevorbereitung und die Kontrolle der Ausgabe verschiedene Sachbearbeitergruppen eingesetzt, für die man aus der Jobstammdatei nach verschiedenen Kriterien (z.B. Firma, Abteilung, Laufhäufigkeiten) Auswertungen erstellen kann.

---

1) Laufen Jobs z.B. in einem strengen 2-Tages-Rhythmus, lassen sie sich durch diesen Aufbau des Terminsteuerworts nicht aussteuern.

Bereitstellungstermin (8)

Die Angabe des Bereitstellungstermins ist für die Ablauf- und Belegungsplanung von großer Wichtigkeit. Er definiert den Zeitpunkt, zu dem die neu abgelochten Daten in geladener Form für die Verarbeitung an der Anlage zur Verfügung stehen. Der Anlieferungstermin der Daten, Loch- und Prüfzeiten der Datenerfassung sowie Kontrollzeiten der Sachbearbeitung bleiben hier unberücksichtigt.

Abgabetermin (9)

Die Angabe des Abgabetermins ist aus planungstechnischen Gründen erforderlich. In den meisten Fällen wird zwar der Morgen des auf den Planungstag folgenden Tages der Fertigstellungstermin des Jobs sein, vor allem im Bereich der Produktionsplanung können aber auch Zwischentermine vorkommen. Als Abgabetermin sei hier der Zeitpunkt definiert, an dem die Jobauswertungen an die Sachbearbeitung zurückgegeben werden müssen. Der Rückgabetermin an die Fachabteilungen ist von untergeordnetem Interesse, da manche Jobs direkt von der Sachbearbeitung erledigt werden[1], und man sonst zusätzlich auch die Termine des Post- und Botendienstes in die Überlegungen einbeziehen müßte.

Ladefiles (10)

Ein Ladefile ist ein Datenträger, auf den die abgelochten Daten vor der Verarbeitung in der Anlage übertragen werden. Man wählt dazu meist ein Magnetband oder eine Magnetplatte. Dieses Vorgehen empfiehlt sich hauptsächlich bei Großrechensystemen, um bei Lese-Operationen des Programms die - gegenüber dem Lochkarten- oder -streifenleser - höhere Lesegeschwindigkeit der Magnetband- und Magnetplattenspeicher ausnutzen zu können.

Verarbeitungszeit (11)

Vor allem für die Belegungsplanung einer Großrechenanlage ist die Verarbeitungszeit eines Jobs eines der wichtigsten Kriterien. Sie könnte auf zweierlei Weise definiert werden: Man kann sie einmal

---

1) Dieser Fall kommt vor allem bei Abstimmungsarbeiten vor, bei denen meist nur ein Vergleich von Soll- und Istzahlen erfolgt.

als Verweilzeit des Jobs im System auffassen und zum anderen als CPU-Benutzungszeit definieren. Die CPU-Zeit stellt in Multiprogramming-Systemen zweifellos das genauere Maß für die Verarbeitungszeit dar, sie ist aber leider für die Zwecke der Belegungsplanung unbrauchbar, denn sie läßt keine Rückschlüsse auf die tatsächliche zeitliche Inanspruchnahme der peripheren Geräte zu. Als Maß für die Verarbeitungszeit muß man daher auf die Verweilzeit des Jobs im System zurückgreifen, und zwar wird man meist einen Mittelwert, der über mehrere Läufe beobachtet wurde, zugrunde legen[1]. Ihre Angabe erfolgt in Minuten.

Druckzeit (12)

Die Druckzeit ist aus zwei Gründen wichtig: a) weil sie für die Einhaltung des Abgabetermins von großer Bedeutung ist und b) weil für jeden Drucker ein bestimmter Kernspeicherplatz zur Verfügung gestellt werden muß. Sie kann eventuell unterteilt werden in eine Druckzeit, die bei der Anlage selbst zum Tragen kommt, weil die Druckausgabe direkt über einen Drucker erfolgt (DZWR), und in eine Druckzeit, die aus dem Drucken von Magnetbändern resultiert (DZDB), die als sog. Druckbänder in einem Job erstellt wurden, z.B. weil sie ein besonderes Papierformat benötigen, das den normalen Druckbetrieb unnötig aufhalten würde. Diese Druckbänder werden meist an einem besonderen Drucker mit einer angeschlossenen Magnetbandeinheit gedruckt. Ihre Angabe erfolgt in Minuten.

zu 3. Maschinenerfordernisse des Jobs

Die in diesen Abschnitt fallenden Daten sind teilweise IBM-spezifisch, da vom Verfasser die Jobstammdatei für eine IBM/370-165 und eine IBM/370-155 aufgebaut wurde. Eine Umformulierung bzw. Ergänzung für Anlagen anderer Hersteller dürfte aber keine Schwierigkeiten bereiten.

---

1) Eine genauere Diskussion der Probleme, die sich für die Belegungsplanung aus der Verarbeitungszeit ergeben, wird in Kapitel 7 vorgenommen.

Spätestens bei der Festlegung der Maschinenerfordernisse muß auch eine Entscheidung getroffen werden, ob die Jobstammdatei step- oder jobbezogen aufgebaut wird. In dieser Arbeit bezieht sich die Belegungsplanung auf den Job als einzuplanende Einheit, und es werden daher bei der Speicherung der Maschinenerfordernisse nur die insgesamt größten Daten der einzelnen Steps berücksichtigt. Der Hauptgrund für die Jobeinplanung ist dabei technischer Natur. In einem Großrechenzentrum werden täglich 150 - 200 Jobs zu erledigen sein, von denen jeder im Durchschnitt aus 5 Steps besteht. Für die Schnelligkeit, mit der eine Belegungsplanung durchgeführt werden kann, ist es aber von ausschlaggebender Bedeutung, ob 150 oder 750 Planungseinheiten verarbeitet werden müssen, da die Kombinationsmöglichkeiten bei der Einplanung mit steigender Zahl der Einheiten stark überproportional anwachsen. Außerdem kommt hinzu, daß ein Operator mit der Steuerung einzelner Steps völlig überfordert wäre, da die Steps im Durchschnitt nur 3-5 Minuten laufen.

Anzahl Steps (13)

Die Anzahl der zu einem Job gehörenden Steps wird bei dem vorliegenden Aufbau nur für statistische Zwecke gespeichert.

Jobpriorität (14)

Die Jobpriorität spielt für die Belegungsplanung eine außerordentlich große Rolle, da das Betriebssystem die Jobs ja ihrer Priorität entsprechend einplant. Die Jobpriorität muß deshalb unter allen Umständen auch bei der Belegungsplanung streng beachtet werden.

Kernspeicherbedarf (15)

Der Kernspeicherbedarf eines Jobs ist abhängig vom Kernspeicherbedarf seines größten Steps. Auch er spielt für die Belegungsplanung eine große Rolle, da nach Zuteilung der übrigen Peripherie letztlich auf Grund des freien Kernspeicherbereichs entschieden wird, ob ein Job laufen kann oder nicht.

Job - Intensität (16)

Die Job-Intensität ist ein Maß für die Ein-/Ausgabe- bzw. Rechenintensität eines Jobs. Sie wird als Quotient aus der CPU-Zeit und

der Verweilzeit eines Jobs ermittelt. Je größer sie ist, desto rechenintensiver ist der Job, d.h. desto weniger Ein-/Ausgabeoperationen werden in ihm vorgenommen. Vor allem in Multiporgramming-Systemen wird der Benutzer versuchen, rechen- und ein-/ausgabeintensive Jobs zu kombinieren, um zu einem möglichst hohen Auslastungsgrad der Zentraleinheit zu kommen.

Anzahl der benutzten Bandeinheiten (17)

Sind in einem Rechenzentrum verschiedene Typen von Bandeinheiten angeschlossen, müssen diese getrennt gespeichert werden (z.B. 7-Spur- und 9-Spur-Einheiten). Besitzt man schnelle und langsame Bandstationen, kann man die Zuordnung von der Jobintensität abhängig machen.

Auch hier muß die Anzahl der maximal in einem Step benutzten Einheiten festgehalten werden, obwohl man sich hinsichtlich der Kombinationsmöglichkeiten dabei Einschränkungen auferlegt, denn diese maximale Anzahl wird voraussichtlich nicht in allen Steps benötigt.

Magnetplattenspeicherbelegung (18)

Für die Plattenspeicher gilt entsprechend das für Magnetbandeinheiten Gesagte. Die Plattenspeicherbelegung ist hier in Zylindern angegeben.

zu 4. Druckerfordernisse

Anzahl der Druckausgabeklassen, Anzahl der Druckbänder (19)

Diese Angaben dienen hauptsächlich wieder Abrechnungs- und Auswertungszwecken, da z.B. durch die Druckausgabeklassen meist das Papierformat und die Anzahl der Durchdrucke bestimmt wird.

Anzahl der erstellten Listen (20)

Die Speicherung der Anzahl der in einem Job erstellten Listen in der Jobstammdatei ist nicht unbedingt notwendig. Sie dient vor allem als Abrechnungsinformation und kann zusätzlich für statistische Auswertungen genutzt werden.

zu 5. K o s t e n - U m l a g e - I n f o r m a t i o n e n (21)

Bei integrierten Programmsystemen kann es vor allem in Konzerngesellschaften häufig vorkommen, daß einzelne Jobs nicht nur für eine, sondern für mehrere Firmen laufen. Die Kosten, die diese Jobs verursachen, sind dann auch den verschiedenen Kostenstellen anzulasten, und die entsprechenden prozentualen Anteile an den Gesamtkosten des Jobs sollten mit den dazugehörigen Kostenstellen in der Jobstammdatei gespeichert werden.

zu 6. I n f o r m a t i o n e n ü b e r F o l g e j o b s (22)

Die Interdependenzen zwischen einzelnen Jobs können leider nicht in vollem Umfang durch eine Gegenüberstellung von Ein-/Ausgabedateien aufgezeigt werden, und es ist daher in bestimmten Fällen[1] notwendig, die Folgejobs in der Jobstammdatei zu speichern, die direkt im Anschluß an einen bestimmten Job laufen müssen.

zu 7. D a t e i - I n f o r m a t i o n e n (23)

Bevor die eigentliche Belegungsplanung durchgeführt wird, ist es notwendig, die logischen Abhängigkeiten der Jobs untereinander festzustellen. Dazu bedient man sich der Ein-/Ausgabedatenbestände (Files), denn wenn z.B. Job A einen File erstellt, der in Job B wieder benutzt wird, so muß Job A ein logischer Vorgänger von Job B sein. Die Speicherung der Dateien in der Jobstammdatei kann sich auf solche Files beschränken, die entweder als Eingabe in den Job oder als Ausgabedatei für einen anderen Job dienen, d.h. Druckbänder und Zwischendateien, die nur innerhalb eines Jobs von einem Step zu einem anderen Step weitergegeben werden, gehören nicht in die Jobstammdatei. Für die einzelnen Files müssen folgende Daten gespeichert werden:

---

1) Eine ausführliche Diskussion dieses Problems erfolgt in Kapitel 4.

1. der Name des Datenbestandes, der als File-Identifikation gilt,
2. die relative Generationsnummer des Datenbestandes und
3. die Dispositionseintragung des Files, die erkennen läßt, ob es sich um einen neu erstellten oder einen bereits bestehenden Datenbestand handelt.

Werden diese Informationen für alle Routinejobs gespeichert, so verfügt man über ein Instrument, mit dessen Hilfe man sowohl einen großen Teil der für ein Großrechenzentrum unerläßlichen statistischen Auswertungen erstellen als auch Planungsaufgaben in vollem Umfang gerecht werden kann.

Natürlich wird der hier vorgeschlagene Satzaufbau je nach Benutzer und Anlagentyp zu modifizieren sein. Der endgültige Aufbau und Inhalt eines Jobstammsatzes wird immer von der Zielrichtung bestimmt sein, die man primär mit der Jobstammdatei verfolgt. Der vom Verfasser vorgeschlagene Aufbau sollte vor allem zur Lösung von Planungsaufgaben beitragen; ein anderer Anwender kann durchaus ein Jobinformationssystem als wichtiger ansehen, aus dem die Cross-Referenzen zwischen Jobs, Jobsteps, Dateien und Datentypen hervorgehen.

Auch das Gebiet der Datenerfassung kann in eine Jobstammdatei aufgenommen werden; in dieser weiteren Ausbaustufe würde sie einen vollständigen Überblick über alle in einem Rechenzentrum anfallenden Aktivitäten ermöglichen[1].

---

1) Auf die Berücksichtigung der Datenerfassung wird in dieser Arbeit verzichtet, obwohl dadurch bestimmte Probleme nur vereinfacht dargestellt werden können.

3.3 Auswertungsmöglichkeiten

Für die in Abschnitt 3.2 besprochene Jobstammdatei bieten sich hauptsächlich drei Auswertungsbereiche an:

1. Erstellung eines Ablaufplans sowie daraus resultierender Belegungspläne,
2. Aufbau eines monatlichen Vorausplanungssystems und
3. Statistische Auswertungen.

Der Aufbau des Ablaufplans wird ausführlich in Kapitel 4 erläutert, so daß hier eine Beschränkung auf die Punkte 2 und 3 erfolgen kann.

zu 2. Aufbau eines monatlichen Vorausplanungssystems

Eine besondere Schwierigkeit für das Management von Rechenzentren liegt in der mangelnden Kenntnis der kumulierten Laufzeiten von Routinejobs für die einzelnen Arbeitstage. Selbst wenn die Lauftermine und Laufhäufigkeiten der einzelnen Jobs frühzeitig bekannt sind, kann und wird sich jeden Monat eine andere Belastung des Rechners ergeben. Teilweise läßt sich dieses Phänomen einfach durch die unterschiedliche Festlegung der Lauftermine nach Kalender- und Arbeitstagen erklären, die in verschiedenen Monaten zu völlig unterschiedlichen Jobzusammenstellungen führen, teilweise kommt es durch die in vielen Rechenzentren nicht geringe Zahl der sog. Abrufjobs zustande, d.h. von Jobs, die sich terminlich nicht im voraus festlegen lassen und die nur auf Abruf der Fachabteilung laufen. Die Erfassung der Abrufjobs ist natürlich in keinem System möglich, und man muß sich bei ihnen mit der Empfehlung begnügen, ihre Zahl so gering wie möglich zu halten. Für alle anderen Jobs läßt sich aber mit Hilfe der Jobstammdatei eine Auswertung erstellen, die einen genauen Aufschluß über die zu erwartenden Rechenzeiten vermittelt. Auf Grund dieser Auswertung ist es dann möglich, Spitzen in der Rechnerbelegung abzubauen, indem einzelne Arbeitsgebiete auf belegungsschwächere Tage verschoben werden. Ein jeweils am Monats-

ende durchgeführter Vergleich der Soll- und Istzeiten läßt voraussichtlich schon nach wenigen Monaten eine gewisse Regelmäßigkeit der Abweichungen erkennen und führt somit zu einer verbesserten Vorausschau[1].

Zum Aufbau eines Monatsplans können nur sehr allgemein gehaltene Hinweise gegeben werden, da hier unternehmenstypische Gesichtspunkte zu beachten sind. Bei der vom Verfasser für ein Großrechenzentrum durchgeführten Monatsplanung[2] haben sich Ausgabelisten, die sowohl tabellarische Tagesübersichten als auch graphische Belastungspläne für die gesamte Planungsperiode ermöglichen, als sehr brauchbar erwiesen. Das Hauptinteresse richtete sich auf die nach Sachgebieten oder Firmen aufgeschlüsselten Verarbeitungs- und Druckzeiten, die zusätzlich jeweils für jeden Tag aufsummiert wurden.

Anhang 1 zeigt den Ausdruck einer Monatsplanungsübersicht für die Routinejobs, die an einem bestimmten Tag auf einer IBM/370-165 zu erledigen waren. Die Übersicht enthält pro Werk (Abteilung) eine Zeile, aus der die Anzahl der Jobs, ihre Verarbeitungsklasse, das Sachgebiet der Jobs[3], die Anzahl der zu ladenden Datenbestände, die Verarbeitungszeit und die Druckzeiten hervorgehen. Pro Tag wird außerdem eine Summenzeile für diese Daten ausgegeben, die eine schnelle Orientierung über die insgesamt vorgeplante Zeit gestattet.

Abb. 9 zeigt einen graphischen Belastungsplan, der Auskunft über die Höhe der Verarbeitungszeiten für einen bestimmten Planungszeitraum gibt. Auf Grund eines solchen Belastungsplans läßt sich frühzeitig erkennen, an welchen Tagen Überbelastungen auftreten, und durch

---

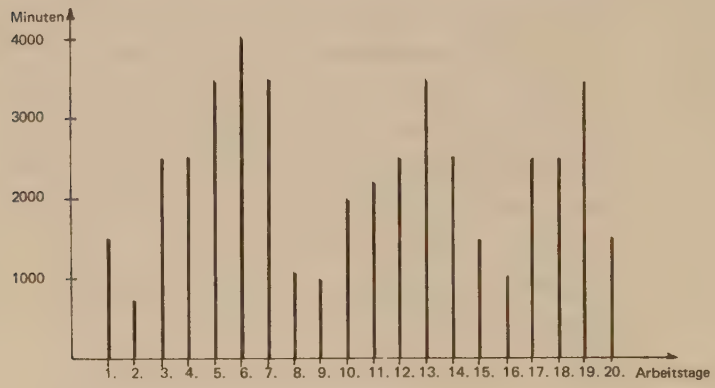
1) Die Bedeutung eines ständigen, möglichst detailliert durchzuführenden Soll-/Istvergleichs kann nicht hoch genug bewertet werden, da nur so die einzelnen Abteilungen zu genauen Angaben gezwungen werden können.

2) Zwei in Fortran IV geschriebene Programme für den Aufbau der Monatspläne von zwei Anlagentypen haben eine Laufzeit von insgesamt etwa 15 - 20 Minuten.

3) Die Ziffern des Werksschlüssels bedeuten dabei beispielsweise: 1: Buchhaltungsjobs; 2: Lohn- und Gehaltsabrechnung; 6: Fertigungssteuerung.

die terminliche Verschiebung einzelner Arbeitsgebiete läßt sich häufig ein den Kapazitätserfordernissen angepaßter Belastungsplan erreichen.

Abb. 9: Verarbeitungszeit-Übersicht für eine Planungsperiode



zu 3. Statistische Auswertungen

Bei den statistischen Auswertungen empfiehlt sich eine Trennung zwischen solchen für Kunden des Rechenzentrums (z.B. Fachabteilungen, Tochtergesellschaften, Fremdfirmen) und internen Abteilungen des Rechenzentrums selbst. Für beide Gruppen lassen sich nämlich inhaltlich völlig verschiedene Auswertungswünsche feststellen.

Die Kunden eines Rechenzentrums richten ihr Hauptinteresse auf die strukturellen Zusammenhänge der Routinejobs, die für sie laufen, d.h. sie sind daran interessiert, die Abhängigkeiten der Jobs untereinander nach Laufhäufigkeiten sortiert, zu erhalten. Diese Jobstrukturen stellen eine wertvolle Entscheidungsunterlage dar, wenn es um eine Nutzen/Kosten-Analyse bestimmter Auswertungen geht. Erzeugt z.B. ein Programm, das gleichzeitig Eingabedateien für verschiedene Folgeprogramme erstellt, zusätzlich Ausgabelisten, so wird man die geringen zusätzlichen Kosten für diese

Listen eher in Kauf nehmen als wenn ein Programm nur eine Liste ausdruckt.

Für die rechenzentrumsinternen Interessenten kommt es dagegen vor allem auf Informationen an, die die Gesamtheit der Jobs betreffen und die manuell nur mit einem erheblichen Aufwand zu gewinnen wären. Von der Vielzahl möglicher Auswertungen sollen hier nur einige besonders wichtige aufgeführt werden, die sich mit Hilfe des in Abschnitt 3.2 vorgeschlagenen Satzaufbaus der Jobstammdatei verwirklichen lassen:

1. Terminsteuerwortauswertung nach Firmen und/oder Sachbearbeitergruppen aufgeschlüsselt. Dabei geht es um die Fragen, wieviele Jobs als Abrufjobs gemeldet sind (Beeinträchtigung der Aussagekraft der Monatsplanung) und ob die Steuerworte logisch sinnvoll ausgefüllt sind.
2. Feststellung der Cross - Referenzen zwischen Jobs und Files. Aus dieser Auswertung geht hervor, welche Datenbestände in einem Job neu erstellt bzw. benutzt werden und von welchen Jobs ein ganz bestimmter Datenbestand benutzt wird.
3. Auswertungen hinsichtlich der Verarbeitungs- und Druckzeiten . Beispielhaft sei hier die in Tab. 6 dargestellte Verteilung der Verarbeitungszeiten aufgezeigt.

Tab. 6: Verteilung der Verarbeitungszeiten von 1707 Routinejobs eines Großrechenzentrums

Jobs mit einer Verarbeitungszeit über	Anzahl der Jobs	Anteil der Jobs
0 Minuten	1707	100 %
10 "	834	48,52 %
20 "	573	33,33 %
30 "	417	24,26 %
45 "	289	16,81 %
60 "	191	11,11 %
120 "	77	4,48 %

4. Auswertungen über die Anzahl der benutzten Druckausgabeklassen . Die Auswertung in Tab. 7 bezieht sich nur auf die über Output-Writer angesprochenen Druckausgabeklassen und nicht auf Druckbänder. Das erklärt die große Anzahl der Jobs, die keine Druckausgabeklassen ansprechen.

Tab. 7: Verteilung der Druckausgabeklassen von 1707 Jobs

Anzahl der benutzten Druckausgabeklassen	Anzahl der Jobs	Anteil der Jobs
0	438	25,7 %
1	1040	60,9 %
2	203	11,9 %
3	21	1,2 %
4	5	0,3 %
Insgesamt	1707	100,0 %

5. Auswertungen über die Anzahl der in einem Job benutzten Eingabedateien bzw. die Anzahl der erstellten Ausgabedateien.
6. Auswertungen über die maximal in einem Job belegte Anzahl von Magnetbandeinheiten bzw. Magnetplattenspeicherplätzen. Für Magnetbandeinheiten zeigt Tab. 8 eine entsprechende Auswertung. Neben der Aufgliederung in Laufzeitklassen ist häufig auch ein Bezug auf die insgesamt von den Jobs erstellten und benutzten Dateien oder der Bezug auf Magnetspeicherklassen für organisatorische Überlegungen von großem Interesse.

Tab. 8: Verteilung der maximal in einem Job belegten Magnetbandeinheiten

Anzahl der maximal belegten Magnetbandeinheiten	Anzahl der Jobs	Anteil	davon Jobs mit einer Laufzeit	
			bis 30 Min.	über 30 Min.
0	56	3,3 %	54	2
1	840	49,1 %	651	189
2	720	42,2 %	450	270
3	88	5,2 %	52	36
≥ 4	3	0,2 %	-	3
Insgesamt	1707	100,0 %	1207	500

Die vielfältigen Auswertungsmöglichkeiten der Jobstammdatei können durch diese Beispiele natürlich nur in begrenztem Umfang angedeutet werden. Dem Benutzer steht mit diesem Instrument aber die Möglichkeit offen, mit geringem Aufwand und in einer verhältnismäßig kurzen Zeit jede gewünschte Auskunft über die gespeicherten Informationen zu erhalten.

4 DURCHFÜHRUNG DER JOBABLAUFPLANUNG FÜR EINE PLANUNGSPERIODE OHNE BERÜCKSICHTIGUNG VON KAPAZITÄTEN

4.1 Die Darstellung von Jobstrukturen mit Hilfe von Netzwerken

Durch den Aufbau der Jobstammdatei sind die Voraussetzungen für die Durchführung einer automatisierten Job-Ablaufplanung geschaffen. In den meisten Rechenzentren werden die Abhängigkeiten zwischen den Routinejobs von den Sachbearbeitern der einzelnen Arbeitsgebiete angegeben. Die Arbeitsvorbereitung hat dann die Aufgabe, in Zusammenarbeit mit der Sachbearbeitung eventuell zusätzlich zu beachtende Schnittstellen zwischen den Sachgebieten herauszufinden und im Ablaufplan zu berücksichtigen. Die für den Multiprogramming-Betrieb wichtigen Parallellaufmöglichkeiten der Jobs werden innerhalb eines Sachgebiets bei Ablauforganisationen dieser Art meist nicht geprüft. Es wird unterstellt, daß nur die Jobs voneinander unabhängiger Sachgebiete parallel zueinander erledigt werden können.

Das vom Verfasser entwickelte Ablaufplanungsverfahren gestattet es, die logischen Abhängigkeiten der Jobs untereinander aufzuzeigen, ihre Reihenfolgen festzulegen und ihre Parallellaufmöglichkeiten sowohl innerhalb eines Sachgebiets als auch bei voneinander unabhängigen Sachgebieten sichtbar zu machen. Das im folgenden dargestellte Verfahren ermöglicht allerdings noch nicht die optimale Steuerung eines Rechenzentrums, denn die Steuerung einer Großrechenanlage setzt die genaue Beachtung der kapazitiven Anforderungen der einzelnen Jobs und ihre Kombination unter Beachtung bestimmter optimierender Regeln voraus. Das Ziel der Ablaufplanung richtet sich zunächst nur auf die Lösung des Reihen-

folgeproblems , das sich für die Routinejobs in jeder Planungsperiode neu stellt. Als Planungszeitraum für den Ablaufplan soll der Arbeitstag gewählt werden. Die Jobs, die an einem Arbeitstag laufen müssen, erhält man durch den Vergleich des für diesen Tag gültigen Terminsteuerwortes mit den Terminsteuerworten sämtlicher in der Jobstammdatei gespeicherter Jobs. Das für einen bestimmten Tag gültige Terminsteuerwort wird dabei jeweils mit Hilfe eines einfachen Programms durch die Abfrage des Maschinendatums aufgebaut. Bei einer Gesamtmenge von 1707 Jobs dauert die Auswahl von 150 - 200 Jobs, die an einem bestimmten Tag zu erledigen waren, nur etwa 30 - 60 Sekunden.

Die Stammsätze der Jobs, die man durch einen Vergleich der Terminsteuerworte für einen Planungszeitraum herausgefunden hat, stehen zunächst aufsteigend nach Jobnamen sortiert zur Verfügung. Die Hauptaufgabe der Ablaufplanung besteht nun darin, eine Methode zu finden, die es erlaubt, durch ein automatisch arbeitendes Verfahren die logischen Abhängigkeiten zwischen den Jobs darzustellen.

Als eine solche Methode zur Bildung von Reihenfolgen bietet sich die Darstellung der Jobstrukturen durch Netzwerke (Graphen) an, denn die Verwendung von Graphen zur Wiedergabe der Jobzusammenhänge ermöglicht eine weitgehende Formalisierung der Ablaufplanung[1]. Zum Verständnis der weiteren Ausführungen sollen zunächst einige Begriffe der Graphentheorie erläutert werden, die von dem ungarischen Mathematiker KÖNIG[2] erstmalig umfassend dargestellt wurde.

Ein Graph ist ein aus einer endlichen oder unendlichen Menge von Knoten und Kanten gebildetes Netzwerk. Die Knoten stellen dabei meist Aktivitäten (Vorgänge) oder Ereignisse dar. In dieser Arbeit sollen den Knoten die einzelnen Jobs entsprechen, die erledigt werden

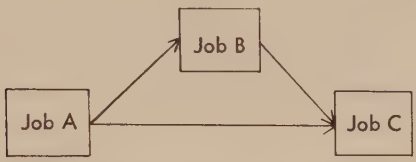
---

1) Die Begriffe "Graph" und "Netzwerk" sollen im folgenden synonym verwendet werden.

2) Dénes KÖNIG, Theorie der endlichen und unendlichen Graphen, New York, 1950.

müssen; die Kanten zeigen die Verbindung zwischen den Knoten auf. Man unterscheidet allgemein gerichtete und ungerichtete Kanten, je nachdem, ob mit ihnen ein Durchlaufsinn angegeben werden soll oder nicht. Bei der Ablaufplanung haben die Kanten die Aufgabe, die Abhängigkeiten zwischen den einzelnen Jobs darzustellen. Sie haben also nur topologische und keine metrischen Funktionen zu erfüllen[1].

Abb. 10: Beispiel eines einfachen Graphen



Die Abb. 10 zeigt einen einfachen, aus drei Knoten bestehenden Graphen. Es ist zu erkennen, daß Job A die beiden Nachfolger B und C besitzt, und Job B nur den Nachfolger C bzw. bei umgekehrter Betrachtung, daß Job C die Vorgänger A und B und Job B den Vorgänger A hat. Die Pfeile geben dabei an, daß Folgejobs erst nach Beendigung des vorangehenden Jobs begonnen werden können.

Der graphischen Darstellung in der Aussagekraft gleichwertig ist eine Darstellung der Abhängigkeiten in Form einer Liste, sie besitzt allerdings nicht die gleiche Anschaulichkeit. Die dem in Abb. 10 dargestellten Beispiel entsprechende Liste würde den in Tabelle 9 wiedergegebenen Aufbau besitzen:

Tab. 9: Beispiel der Darstellung eines Graphen in Listenform

Vorgänger	Nachfolger
Job A	Job B, Job C
Job B	Job C
Job C	keine

oder

Nachfolger	Vorgänger
Job C	Job A, Job B
Job B	Job A
Job A	keine

1) Im Rahmen der Netzplantechnik - dem wohl bekanntesten Anwendungsgebiet der Graphentheorie - haben die Kanten teilweise auch metrische Funktionen zu übernehmen. Darauf soll hier aber nicht näher eingegangen werden.

Ein Graph mit n Knoten kann maximal $\binom{n}{2} = \frac{n}{2}(n-1)$ Kanten[1] enthalten, wobei dann jeder Knoten direkt mit jedem anderen Knoten verbunden ist (vollständiger Graph), und er besitzt minimal (n - 1) Kanten[2], wenn von jedem Knoten nur eine Kante bzw. eine Folge von aufeinanderfolgenden Kanten zu jedem anderen Knoten führt. Graphen mit der zuletzt beschriebenen Eigenschaft bezeichnet man auch als Baum.

Abb. 11: Beispiel eines vollständigen Graphen

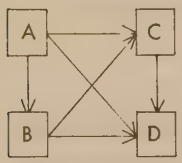
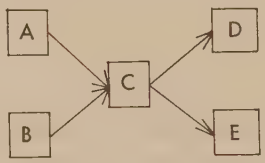


Abb. 12: Beispiel eines Baums



Ein Graph mit gerichteten Kanten wird allgemein als gerichteter Graph bezeichnet. In der Ablaufplanung, bei der die Jobs als Knoten und ihre Abhängigkeiten als gerichtete Kanten aufgefaßt werden, hat man es also mit gerichteten Graphen zu tun.

Neben der graphischen und listenmäßigen Darstellung von Jobstrukturen hat sich die Darstellung von Graphen vor allem auch in Matrixschreibweise bewährt. Bei der Aufstellung des Ablaufplans mit Hilfe elektronischer Datenverarbeitungsanlagen ist diese Darstellungsart außerdem mit einem relativ geringen Aufwand an Speicherplatz verbunden. Bei der Matrixdarstellung von Netzwerken benutzt man eine quadratische Matrix, bei der die Zahl der Zeilen bzw. die der Spalten gleich der Anzahl der Vorgänge ist. Üblicherweise trägt man die Vorgänger in den Zeilen

---

1) vgl. dazu H. MÜLLER-MERBACH, Optimale Reihenfolgen, Berlin 1970, S. 7

2) vgl. dazu H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 7

und die Nachfolger in den Spalten ein. Die Kennzeichnung der Abhängigkeiten erfolgt durch eine (0; 1) Markierung[1]. Besitzt ein Vorgang I einen Folge-Vorgang J, wird an der entsprechenden Stelle der Matrix eine "1" eingetragen; existiert keine Abhängigkeit zwischen den Vorgängen, erhält die entsprechende Stelle eine "0". Für das in der Abb. 10 bzw. Tab. 9 dargestellte Beispiel bekommt man folgende Matrix:

Tab. 10: Beispiel der Matrixdarstellung eines Graphen

Vorgänger \ Nachfolger	Job A	Job B	Job C
Job A	0	1	1
Job B	0	0	1
Job C	0	0	0

Das Auffinden und Darstellen logischer Abhängigkeiten mit Hilfe von Netzwerken wird vor allem bei den Programmen für die Durchrechnung von Netzplänen seit längerer Zeit benutzt. Die Anwendung dieses Verfahrens zur Schaffung der Voraussetzungen für die Steuerung und Einsatzplanung von Großrechensystemen ist dem Verfasser hingegen weder in der Praxis noch in der Literatur bisher bekanntgeworden.

BUDIL[2] bezeichnet den ersten Entwurf eines Netzwerks für die Abbildung einer Projektstruktur zu Recht als den schwierigsten und aufwendigsten Teil der Ablaufplanung. Dies gilt jedoch nur dann, wenn die Aufgabe des Planenden sich sowohl auf die Definition der Vorgänge einschließlich der vorzunehmenden Zeitschätzungen als auch auf die Festlegung der Abhängig-

---

1) Diese Markierungsform gestattet es, bei einer Byte-orientierten EDV-Anlage einzelne Bits zu setzen bzw. nicht zu setzen. Daraus resultiert eine besonders günstige Kernspeicherausnutzung.

2) W. BUDIL, Netzplantechnik - ein neues System zur wirkungsvollen Projekt- und Terminplanung, Zeitschrift für Organisation 34,1965, S.42.

keiten zwischen den einzelnen Vorgängen bezieht. Die Abhängigkeiten sind bei vielen Problemstellungen teilweise auf Grund technischer Bedingungen bindend vorgegeben, unterliegen teilweise aber auch nicht-technischen Voraussetzungen wie z.B. kapazitiven Beschränkungen und können vom Planer in diesem Fall frei vorgegeben werden. Beide Punkte, nämlich die Definition der Vorgänge und die Festlegung der Abhängigkeiten begründen die Schwierigkeiten, die bei der Aufstellung von Netzwerken bestehen. Nur selten ist der Planer bei verschiedenen Projekten in der glücklichen Situation, auch nur annähernd die gleichen Vorgänge mit gleich großen Dauern und unveränderten Abhängigkeiten vorzufinden.

Dieses Problem stellt sich prinzipiell auch bei der Ablaufplanung in einem Rechenzentrum, denn die meisten Jobs wechseln von Tag zu Tag, ihre Dauern ändern sich im Monats- oder Jahresverlauf nicht unbeträchtlich, und auch die Abhängigkeiten müssen täglich neu festgestellt oder zumindest überprüft werden.

Durch die Jobstammdatei ist in diesem Fall jedoch die Gesamtmenge der Jobs bekannt, die auf der Anlage laufen müssen, und durch die Auswertung des Terminsteuerworts ist es auf eine sehr einfache Art möglich, die an einem bestimmten Tag zu erledigende Teilmenge von Jobs festzustellen. Die einzelnen Vorgänge (Jobs) brauchen bei diesem Verfahren also nicht täglich neu definiert zu werden, und die Dauern der Jobs sind ebenfalls aus der Jobstammdatei bekannt[1]. Wenn es zusätzlich gelingt, die Abhängigkeiten automatisch festzustellen, kann ohne großen manuellen Aufwand täglich das Netzwerk der zu erledigenden Jobs aufgestellt werden.

Die für die Datenverarbeitung relevanten Abhängigkeiten zwischen den einzelnen Jobs ergeben sich auf Grund der Dateien, die in die Jobs hineinlaufen und der Dateien, die in ihnen erstellt werden. Neben diesen rein datenverarbeitungsbezogenen Abhängigkeiten können darüber hinaus sach-

---

1) Eine eingehende Betrachtung des Problems der Dauerschätzung von Jobs (Verweilzeiten) wird in Kapitel 7 vorgenommen. Über den Änderungsdienst ist sicherzustellen, daß die jeweils gültige Dauer gespeichert ist.

liche Abhängigkeiten bestehen, die beispielsweise auf Abstimmungsvorgängen beruhen. In solchen Fällen kann es notwendig sein, einen Job oder auch eine Folge von Jobs solange zu wiederholen, bis das gewünschte Abstimmungsergebnis vorliegt. Solche Abhängigkeiten kann eine Ablaufplanung, die von den Informationen der Jobstammdatei ausgehen muß, nicht berücksichtigen, da sie stets vom Ergebnis des am Planungstag durchgeführten Laufs abhängig und zum Planungszeitpunkt unbekannt sind.

Die Berücksichtigung der anderen Abhängigkeiten, die man auch als logische Jobabhängigkeiten bezeichnen könnte, ist dagegen stets bis zum Planungszeitpunkt möglich.

Da Jobfolgen durch die Dateiweitergabe zwischen voneinander abhängigen Jobs begründet werden, muß die automatisierte Festlegung dieser Abhängigkeiten von den in den einzelnen Betriebssystemen möglichen Dateibezeichnungen ausgehen. Wie in Kapitel 2 bereits beschrieben, gestatten es die Data Management Routinen der komfortableren Betriebssysteme, Dateien auf verschiedene Arten zu beschreiben und zu definieren. Im folgenden soll nur an Hand der für die Ablaufplanung günstigsten Methode der Dateibezeichnung, bei der eine Datei durch ihren Namen, ihre relative Generationsnummer und ihre Dispositionsangabe eindeutig festgelegt ist, die Feststellung der Abhängigkeiten aufgezeigt werden. Alle anderen Methoden erfordern einen wesentlich höheren Aufwand bei der Pflege der Jobstammdatei und machen dadurch das gesamte System der Ablaufplanung ineffizienter.

Jobabhängigkeiten können durch Dateien begründet werden, die sich auf Magnetbändern befinden und von denen mehrere Generationen erstellt werden, und sie existieren außerdem auf Grund von Dateien, die sich auf festen Magnetplattenspeicherbereichen befinden. Die Identifikation der Dateien erfolgt durch den alphanumerischen Dateinamen und die absolute oder relative Generationsangabe. Die Verbindung zu dem physikalischen Datenträger (Magnetband oder Magnetplatte) wird durch den

sog. "File-Katalog" gewährleistet, in dem vermerkt ist, welche Bandrolle zu welcher Datei gehört[1]. Da die Ermittlung der Jobabhängigkeiten auf Grund von Magnetbanddateien und die Ermittlung auf Grund von Magnetplattendateien in unterschiedlicher Weise vorgenommen werden muß, sollen zunächst die Jobfolgen erläutert werden, die sich an Hand der auf Magnetbändern gespeicherten Dateien ergeben.

Beispiel: Von der Datei F 10100 müssen drei Generationen verfügbar sein. Der File-Katalog enthält dann in zwei aufeinanderfolgenden Zeitpunkten A und B folgende Angaben:

Tab. 11: File-Katalogeintragung für eine Magnetbanddatei

Filename	absolute Generationsnummer		relative Generationsnummer		Rollennummer
	Zeitp. A	Zeitp. B	Zeitp. A	Zeitp. B	
F 10 100	27	-	- 2		2830
F 10 100	28	28	- 1	- 2	3546
F 10 100	29	29	0	- 1	1016
F 10 100		30		0	4711

Im Zeitpunkt A sollen die Generationen 27 - 29 der Datei aufbewahrt werden.

Wird in einem Job zum Zeitpunkt B von dieser Datei nun eine neue Generation, d.h. die Generation 30, erstellt, fällt die älteste Generation 27 aus dem Katalog heraus, d.h. die Magnetbandrolle 2830 steht wieder zu einer neuen Beschriftung zur Verfügung und die auf der Magnetbandrolle 4711 neu erstellte Datei wird zusätzlich als Generation 30 in den Katalog aufgenommen. Die relativen Generationsnummern ändern sich entsprechend den Angaben in Tab. 11. Im Operating System der IBM wird die "jüngste" bestehende Generation einer Datei immer mit

1) s. Kap. 2.3, S. 32.

der relativen Generationsnummer (0) im Katalog geführt, sie wird jedoch stets in einem Job mit der relativen Generationsnummer (+ 1) erstellt. Generation 30 ist entsprechend in dem zugehörigen Job mit der relativen Generation (+ 1) erstellt worden, wird aber als relative Generation (0) in den Katalog gebracht. Im Katalog erhält die neueste Generation also immer die relative Generation (0). Werden in einem einzelnen Job zwei neue Generationen einer Datei erstellt, so geschieht dies in dem Job mit den relativen Generationen (+1) und (+2). Im Katalog wird (+1) dann zu (-1) und (+2) zu (0). Der nächste Job, der die mit (+2) erstellte Generation benutzt, holt sie sich mit (0) aus dem Katalog.

Bei dieser Art der Bezeichnung können Eingabedateien grundsätzlich nur relative Generationsnummern tragen, die ≤ 0 sind. Die relativen Generationsnummern von Ausgabedateien sind dagegen stets > 0. In Verbindung mit dem Dateinamen erlaubt diese Festlegung eine automatische Bestimmung von Vor- und/oder Folgejobs.

Tab. 12: Automatische Bestimmung von Vor- und Folgejobs

Job	Eingabedatei	Ausgabedatei
A 38	F 10 048 (0) F 30 400 (0)	F 30 410 (+ 1)
A 39	F 60 077 (0)	F 60 077 (+ 1)
A 50	F 10 100 (0) F 20 532 (0) F 10 048 (0)	F 10 110 (+ 1) F 20 532 (+ 1) F 10 048 (+ 1)
A 110	F 10 110 (0) F 20 532 (0)	F 60 077 (+ 1) F 30 400 (+ 1)

Die in Tabelle 12 aufgeführten Jobs A 38, A 39, A 50 und A 110 lesen die Dateien F 10048, F 20532, F 30400, usw.; sie schreiben die Dateien F 60077, F 10048 und F 20532 fort[1], und sie geben die noch nicht existierenden Dateien F 30410, F 10110, F 60077 und F 30400 aus.

1) Von Fortschreibung einer Datei spricht man nur, wenn die Datei in einem Job gelesen und von ihr im gleichen Job eine neue Generation erstellt wird.

Vergleicht man die Dateinamen und die relativen Generationen der Ausgabedateien mit denen der Eingabedateien, so erhält man eine eindeutige Reihenfolge, in der die Jobs erledigt werden müssen.

Job A 50 erstellt die Datei F 10 110, Job A 110 benötigt diese Datei, d.h. A 50 muß vor A 110 laufen. Dieselbe Jobfolge erhält man bei einer Betrachtung der Datei F 20 532. Die in A 50 erstellte Datei F 10 048 (+1) wird in A 38 benötigt, d.h. die Reihenfolge lautet hier: A 50 $\Rightarrow$ A 38. Aus den Dateien von Job A 110 ergeben sich die Reihenfolgen:

A 110 $\Rightarrow$ A 39

A 110 $\Rightarrow$ A 38.

Die Jobs A 38 und A 39 besitzen keine Folgejobs, da ihre Ausgabedateien in keinem der aufgeführten Jobs als Eingabe benötigt werden. Die endgültige Reihenfolge lautet demnach:

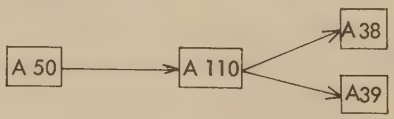
Job A 110 ist Folgejob von Job A 50

Job A 38 und Job A 39 sind Folgejobs von Job A 110 und

damit ist auch Job A 38 Folgejob von Job A 50.

Mit Hilfe eines Netzwerks kann man diese Abhängigkeiten recht anschaulich darstellen, wie das in Abbildung 13 gezeigt wird:

Abb. 13 : Darstellung der in Tab. 12 dargestellten Jobabhängigkeiten als Netzwerk



Es ist hier zwar auch eine andere Reihenfolge möglich, wenn man z.B. in dem Job A 38 aus anwendungsbezogenen Gründen nicht die in A 110 neu zu erstellende Generation der Datei F 30 400 benutzen kann, sondern die vorher erstellte Generation benötigt wird. Untersuchungen haben aber ergeben, daß bei Jobs, die an einem Tag erledigt werden müssen, in der überwiegenden Zahl der Fälle die Reihenfolge eingehalten wird, die sich auf

Grund der Vergleiche zwischen den Ein- und Ausgabedatenbeständen ergibt. Ständig wiederkehrende Abweichungen von dieser automatisch festgelegten Reihenfolge sollten in dem Feld "Folgejobs" der Jobstammdatei gespeichert werden. Enthält dieses Feld eine Eintragung, so sollte die Dateistruktur für diesen Job nicht untersucht und es sollten nur die angegebenen Folgejobs berücksichtigt werden.

In dem auf den Seiten 79 und 80 besprochenen Beispiel wurden die Folgejobs eines bestimmten Jobs festgestellt. Genauso ist auch die Bestimmung der Vorjobs möglich. Dazu werden die Eingabedateien eines Jobs mit den Ausgabedateien aller anderen Jobs verglichen. Stellt man dabei fest, daß Job A eine Datei benötigt, die in Job B erstellt oder verändert worden ist, so kann man sagen, daß Job B ein Vorjob von Job A ist.

Jobabhängigkeiten können nicht nur durch auf Magnetbändern gespeicherte Dateien, sondern auch durch Dateien gegeben sein, die auf Magnetplatten gespeichert sind. Von Magnetplattendateien existiert grundsätzlich nur eine, nämlich die jeweils jüngste Generation. Die Bestimmung von Folgejobs kann in diesem Fall nicht in der geschilderten Form durchgeführt werden. Einen Anhaltspunkt für die Reihenfolge der Jobs bietet hier die Dispositionsangabe der Datei, in der angegeben wird, ob es sich um eine bereits erstellte (OLD), eine zu ändernde (MOD), eine neu zu erstellende (NEW), eine von mehreren Jobs gleichzeitig zu benutzende (SHR) bzw. eine zu katalogisierende Datei (CATLG) handelt. Bei Plattendateien handelt es sich meist um sog. Stammdateien[1], deren Einzelinformationen in vielen Jobs benötigt, die aber nur relativ selten verändert werden. Jobs, die die Datei nur lesen und nicht verändern wollen, werden sie grundsätzlich mit der Disposition "SHARE" ansprechen, da sie bei dieser Angabe mehreren Jobs, die parallel zueinander ablaufen, nacheinander zur Verfügung steht, d.h. nicht vom ersten Job, der sie anspricht, für die Dauer seiner Verweilzeit für die anderen Jobs gesperrt wird. Jobs, die sie verändern wollen, benutzen dagegen die Dispositionsangabe OLD, d.h. diese Jobs sperren sie während ihrer Dauer für andere Jobs.

1) Stammdateien sind z.B. die Personalstammdatei, die Lohnstammdatei, Tabellendateien usw.

Wird dieses Prinzip der Dispositionsangabe bei allen Jobs streng eingehalten, kann man im allgemeinen davon ausgehen, daß zunächst der Job laufen muß, der die Dispositionsangabe OLD enthält und daß erst nach seiner Beendigung die Jobs mit der Dispositionsangabe SHARE erledigt werden dürfen. Diese Jobs können dann, obwohl sie den gleichen Eingabedatenbestand benutzen, parallel zueinander ablaufen. Bei Banddateien, die nicht die bei Plattenspeichern gegebene Möglichkeit des direkten Zugriffs kennen und Dateien nur sequentiell verarbeiten können, ist dies aus technischen Gründen nicht möglich, sie können stets nur von einem Job in Anspruch genommen werden.

Das Problem der Feststellung von Abhängigkeiten bei Jobs, die Plattendateien benutzen, hat sich in der Praxis bisher kaum gestellt, da diese Jobs fast immer zusätzlich auch Banddateien benötigten, die eine eindeutige Festlegung der Abhängigkeiten zuließen. In Zweifelsfällen kann aber auch hier das Feld "Folgejobs" in der Jobstammdatei ausgefüllt und so die gültige Jobfolge festgelegt werden.

4.2 Rangrechnung und Blockbildung

Der Vergleich zwischen den Ein- und Ausgabedateien der Jobs ergibt jeweils nur für einen Job die Vor- bzw. Folgejobs. Bei einer kleinen Gesamtzahl von Jobs können aus diesen Einzelinformationen leicht die Abhängigkeiten aller Jobs untereinander bestimmt werden. Schon von etwa 30 Jobs an wird der Aufwand für die Bestimmung der gesamten Folge jedoch sehr hoch. Die Bestimmung der Gesamtfolge ist aber unbedingt erforderlich, da es ja darauf ankommt, dem Maschinenbedienungspersonal sagen zu können, welche Jobs als erste erledigt werden müssen, welche Jobs auf diesen aufbauen usw. Dieses topologische Problem kann mit Hilfe der sog. "Rangrechnung" gelöst werden. Die Rangrechnung ordnet jedem Job eine Ziffer (seinen "Rang") zu, von der sich ablesen läßt, wieviele Vorjobs er mindestens besitzt. Ein Job im Rang 0 besitzt keinen Vorjob, ein Job im Rang 1 besitzt mindestens einen Vorjob, usw. Wären die Jobs der Reihenfolge entsprechend aufsteigend durchnumeriert,

würde sich diese Rangrechnung erübrigen, da ein Vorjob zu einem bestimmten Job dann immer einen niedrigeren Jobnamen besäße als der betrachtete Job. Bei den häufig wechselnden Abhängigkeiten ist eine aufsteigende Numerierung der Jobs in einem Rechenzentrum jedoch nicht möglich. Es kommt hinzu, daß der Jobname häufig Informationen beinhaltet, die eine willkürliche Numerierung verbieten (z.B. Abrechnungsinformationen).

Zur Rangrechnung bieten sich mehrere Verfahren an, die sich sowohl in der Methode als auch in der Rechenzeit voneinander unterscheiden. Hier in diesem Abschnitt 4.2 sollen zwei Verfahren vorgestellt werden, die hinsichtlich des benötigten Kernspeicherplatzes und der Rechenzeit stark differieren. Beide Verfahren sollen am gleichen, einfachen Beispiel erläutert werden: Gegeben seien 10 Jobs, bei denen der Vergleich der Ein- und Ausgabedateien die in Tab. 13 dargestellten Abhängigkeiten ergeben habe.

Tab. 13: Beispiel zur Rangrechnung

Job-Nr.	Folgejobs
1	4,5
2	5
3	6
4	7
5	7
6	8
7	9
8	9,10
9	-
10	-

Rangrechnungs - Verfahren 1 [1]:

Das erste Verfahren der Rangermittlung geht von einer quadratischen Matrix aus, bei der die Anzahl der Zeilen und die der Spalten gleich der Anzahl der Jobs ist. In dieser Matrix werden diejenigen Elemente gleich 1 gesetzt, denen Abhängigkeiten zwischen Jobs entsprechen. Die anderen Matrixelemente sind gleich 0 zu setzen. Tab. 14 stellt die Matrix zu dem in Tab. 13 dargestellten Beispiel dar.

Tab. 14 : Matrix zu Rangrechnungsverfahren 1

Vorjobs i \ Folgejobs j	1	2	3	4	5	6	7	8	9	10
1	0	0	0	1	1	0	0	0	0	0
2	0	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	1	0	0	0	0
4	0	0	0	0	0	0	1	0	0	0
5	0	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0
Zahl der Vorjobs										
vor dem 1.Durchlauf	0	0	0	1	2	1	2	1	2	1
nach dem 1.Durchlauf	-	-	-	0	0	0	2	1	2	1
nach dem 2.Durchlauf				-	-	-	0	0	2	1
nach dem 3.Durchlauf							-	-	0	0

Man bildet zunächst die Spaltensummen, die Auskunft darüber geben, wieviele Vorjobs ein Job der Spalte j (j=1,...,n) besitzt. Jobs, deren Spaltensumme gleich 0 ist, besitzen keinen Vorjob, d.h. sie sind die Startjobs einer Jobfolge und erhalten den Rang 0 zugeordnet. Nun werden in der Matrix die mit 1 gekennzeichneten Elemente der Z e i l e n i gestrichen, deren Jobs i n diesem 1. Durchlauf einen Rang erhalten

1) H. GÖTZKE , Netzplantechnik, Leipzig 1969, S. 50 ff

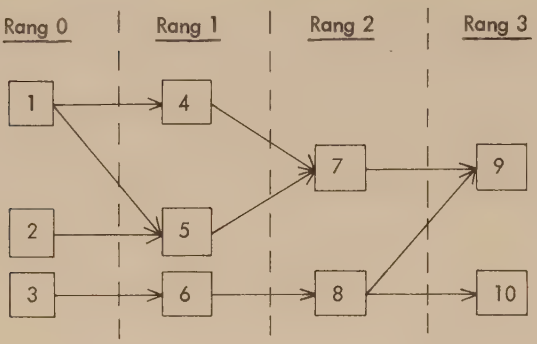
haben. Im Beispiel werden also die Elemente in Zeile 1 (1; 4), (1; 5), in Zeile 2 (2; 5) und in Zeile 3 (3; 6) gestrichen, weil den Jobs 1, 2 und 3 der Rang 0 zugeteilt worden ist. Die Spaltensummen werden anschließend neu gebildet und die Jobs, deren Spaltensumme n a c h diesem ersten Durchlauf gleich 0 ist, erhalten einen um 1 höheren Rang. Im Beispiel sind das die Jobs 4, 5 und 6, die damit Rang 1 besitzen. Wieder werden in den entsprechenden Zeilen (4, 5, 6) die von 0 verschiedenen Elemente gestrichen, die neuen Spaltensummen werden gebildet usw., bis alle Spaltensummen 0 ergeben und allen Jobs ein Rang zugeteilt wurde. Das Ergebnis der Rangrechnung für das in Tab. 13 angegebene Beispiel ist in Tab. 15 dargestellt.

Tab. 15 : Ergebnis der Rangrechnung für das Beispiel in Tab. 13

Rang	Job-Nr.
0	1, 2, 3
1	4, 5, 6
2	7, 8
3	9, 10

Durch dieses Verfahren sind die 10 Jobs in 4 Rangklassen aufgeteilt worden, die eine sehr rasche Orientierung über die Jobzusammenhänge zulassen und die auch die graphische Darstellung sehr erleichtern:

Abb. 14 : Netzwerk des Beispiels aus Tab. 13 mit Rangeinteilung



Bei der vom Verfasser durchgeführten Programmierung dieses speicheraufwendigen und relativ langsam arbeitenden Verfahrens hat es sich aus programmtechnischen Gründen als günstig erwiesen, wenn man bei der Rangermittlung nicht mit den echten Jobnamen, sondern mit intern vergebenen Jobnamen arbeitet, die über einen Vektor mit den echten Jobnamen verknüpft werden können. Auch das zweite Verfahren arbeitet mit diesen internen Jobnamen.

Rangrechnungs - Verfahren 2 [1]:

Ausgangspunkt für das Rangrechnungs-Verfahren 2 ist eine Matrix (Liste), bei der zu einem betrachteten Job nur die direkten Folgejobs angegeben zu werden brauchen.

---

1) H. WILLE, K. GEWALD, H.D. WEBER, Netzplantechnik, Bd. 1: Zeitplanung, 2. Auflage, München 1967, S. 138 f.

Tab. 16: Tabelle zur Rangrechnung mit Hilfe des 2. Verfahrens

Job-Nr.	Folgejobs	Rang				
		0	1	2	3	4
1	4, 5	0				
2	5	0				
3	6	0				
4	7	~~0~~	1			
5	7	~~0~~	1			
6	8	~~0~~	1			
7	9	~~0~~	~~1~~	2		
8	9, 10	~~0~~	~~1~~	2		
9	-	~~0~~	~~1~~	~~2~~	3	
10	-	~~0~~	~~1~~	~~2~~	3	

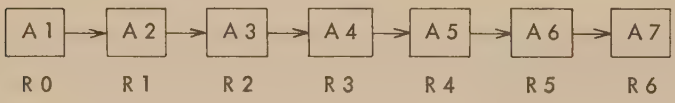
Bei diesem Verfahren ordnet man zunächst jedem Job einen provisorischen Rang 0 zu. Dann wählt man den ersten Job der Liste aus und prüft, ob seine Folgejobs einen um mindestens 1 höheren Rang besitzen als er selbst. Ist das nicht der Fall, erhöht man die provisorischen Ränge der Folgejobs solange, bis diese Bedingung erfüllt ist. Die Folgejobs von Job 1 würden also den Rang 1 erhalten. Dann wird der nächste Job der Liste ausgewählt (Job 2) und für ihn geprüft, ob seine Folgejobs einen um 1 höheren Rang besitzen als er ihn besitzt. Die Bedingung ist in diesem Fall erfüllt, da Job 5 bereits den Rang 1 besitzt. Man wählt wieder den nächsten Job aus, usw., bis alle Jobs durchlaufen sind. Haben sich beim Abarbeiten der Liste Rangänderungen ergeben, beginnt man wieder beim ersten Job der Matrix (Job 1) und führt das gesamte Verfahren erneut durch. Dies geschieht solange, bis sich bei einem gesamten Durchlaufen der Liste keine Rangänderungen mehr ergeben. Die zunächst provisorisch festgesetzten Ränge entsprechen dann den tatsächlichen Rängen der Jobs.

Dieses Verfahren arbeitet in der Regel wesentlich schneller als das Rangrechnungs-Verfahren 1. Außerdem benötigt es bei weitem nicht so viel Kernspeicherplatz wie das 1. Verfahren. In diesem Beispiel stehen bereits nach zweimaligem Durchlaufen der Jobs die Ränge aller Jobs fest. Das

hängt hier allerdings mit der Jobnumerierung zusammen, da die Nummer eines Jobs immer kleiner ist als die Nummer eines Folgejobs.

Wie ohne weiteres einsichtig ist, kann der Wert der Rangziffer maximal gleich der um 1 verminderten Anzahl der Jobs sein, und zwar gilt dies, wenn nur ein Startjob existiert und jeder Job nur einen Folgejob hat. Je stärker die Vermaschung der Jobstruktur ist, desto geringer ist die Zahl der Ränge.

Abb. 15 : Beispiel zur Veranschaulichung der maximal erreichbaren Rangziffer bei 7 Jobs (A 1, ..., A 7) und den Rängen (R 0, ..., R 6)

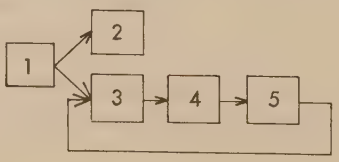


Betrachtet man das in Tab. 17 dargestellte Beispiel und wendet darauf das Rangrechnungs-Verfahren 2 an, so stellt man fest, daß die Anzahl der Durchläufe durch die Jobs größer werden kann als die um 1 verminderte Anzahl der Jobs.

Tab. 17 : Beispiel zur Bestimmung der Zahl der Durchläufe

Job	Folgejobs
1	2, 3
2	-
3	4
4	5
5	3

Abb. 16 : Graph der Tab. 17



Bedingt durch die Abhängigkeiten (3, 4), (4, 5), (5, 3) kommt das Verfahren zu keinem Ende. Job 1 wird zwar richtig der Rang 0 zugeordnet und Job 2 der Rang 1, die Jobs 3, 4, 5 erhalten dagegen eine ständig an-

steigende Rangziffer, falls das Verfahren nicht abbricht, wenn die Rangziffer gleich der Anzahl der Jobs ist. Abhängigkeiten, die solche Rückverweisungen enthalten, werden allgemein als Schleifen oder Zyklen bezeichnet. Da bei dem Vorliegen von Zyklen eine eindeutige Bestimmung der Reihenfolge für die Erledigung der Jobs nicht mehr möglich ist, besteht eine der Hauptaufgaben der Rangrechnung in der Aufdeckung dieser Zyklen.

In der Job-Ablaufplanung treten Zyklen bei etwa 10 % der Jobs auf. Allerdings handelt es sich hierbei nur um eine unechte Zyklusbildung, die auf Grund der Ein- und Ausgabedateien zustande kommt. Der Sachbearbeitung bzw. der Arbeitsplanung eines Rechenzentrums ist es in allen Fällen möglich, die gültige Reihenfolge anzugeben, und diese kann bei ständig wiederkehrenden Zyklen über das Feld "Folgejobs" in der Jobstammdatei verankert werden.

Das Zustandekommen von Schleifen soll an dem in Tab. 18 dargestellten Beispiel erläutert werden.

Tab. 18 : Beispiel zur Zyklusbildung

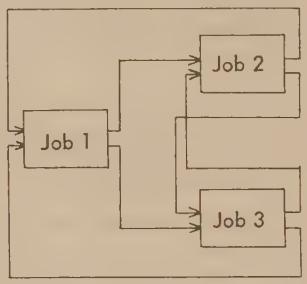
Job-Nr.	Eingabedateien	Ausgabedateien
1	F 10 100 (0) F 20 200 (0)	F 10 100 (+ 1) F 20 201 (+ 1)
2	F 10 100 (0) F 20 201 (0)	F 10 100 (+ 1) F 20 201 (+1)
3	F 20 200 (0) F 20 201 (0)	F 20 200 (+ 1) F 20 201 (+ 1)

Durch einen Vergleich der Ein-/Ausgabedateien kommt man zu folgenden Abhängigkeiten:

Tab. 19 : Feststellung der Abhängigkeiten für das Beispiel aus Tab. 18

Job-Nr.	Folgejobs
1	2,3
2	1,3
3	1,2

Abb. 17 : Graph der Tab. 19



Jeder Job hat die beiden anderen der insgesamt drei Jobs jeweils als Folgejobs, d.h. die Zuordnung von Rängen ist in diesem Fall nicht möglich, da diese Jobs Schleifen bilden. Die Auflösung dieser Schleifen muß von der Arbeitsvorbereitung bzw. der Sachbearbeitung durchgeführt werden und bereitet in der Praxis auch meist keine Schwierigkeiten. Problematisch ist dagegen die manuelle Feststellung solcher Zyklen aus 150 - 200 Jobs, die an einem Tag zu verarbeiten sind. Die Kenntlichmachung von Zyklen ist mit Hilfe eines an die Rangrechnung anschließenden Verfahrens[1] möglich, das einsetzt, sobald ein Rang gleich der Anzahl der einzplanenden Jobs wird. Dieses Verfahren soll im folgenden kurz an Hand des Beispiels aus Tab. 18 beschrieben werden. Es läßt sich ohne Schwierigkeiten auch auf Jobstrukturen anwenden, die mehrere, voneinander unabhängige Zyklen beinhalten.

---

1) vgl. dazu WILLE, GEWALD, WEBER: Zeitplanung, a.a.O., S. 140 f.

Ausgangspunkt des Verfahrens zur Zyklussuche ist die bei Rangrechnungs-Verfahren 2 zur Rangbestimmung benutzte Matrix[1], in der zu jedem Job die Folgejobs sowie ihre provisorischen Ränge angegeben sind. Aus dieser Matrix werden die Jobs ausgewählt, deren provisorische Rangziffer gleich der Anzahl der Jobs ist, die also einen Zyklus oder mehrere Zyklen bilden. Diese Jobs werden mit ihren Folgejobs in einer dem Rangrechnungs-Verfahren 2 entsprechenden Matrix übernommen. In der ersten Stufe des Verfahrens wird allen Jobs der Matrix eine Markierung 0 zugeordnet. Dann setzt man die Markierung des ersten Jobs der Matrix auf "1" und prüft für den nächsten Job der Matrix, ob die Markierung eines seiner Folgejobs gleich "1" ist. Ist das der Fall, setzt man die Markierung des betreffenden Jobs ebenfalls auf "1" und fährt mit der Überprüfung des nächsten Jobs fort, usw. Ist die Markierung der Folgejobs ungleich "1", überprüft man ebenfalls den nächsten Job. Im Beispiel würde zunächst Job 1 die Markierung "1" erhalten. Da Job 1 Folgejob von Job 2, dem nächsten Job der Liste ist, erhält Job 2 ebenfalls die Markierung "1". Entsprechend verfährt man bei Job 3. Haben sich nach dem vollständigen Durchlaufen der Matrix Markierungsänderungen ergeben, d.h. hat man außer bei dem ersten Job der Liste noch bei einem andern Job eine Markierung "1" vorgenommen, führt man diese erste Stufe des Verfahrens solange erneut für alle Jobs durch, bis sich keine Markierungsänderungen mehr ergeben.

Hat man bisher die Jobs, die zu einem Zyklus gehören, auf indirektem Wege gekennzeichnet, indem man bei jedem Job geprüft hat, ob bei einem seiner Folgejobs die Markierung "1" gesetzt war, bevor er selbst diese Markierung erhielt, so führt man nun in einer zweiten Stufe des Verfahrens eine direkte Markierung durch. Dazu erhält der erste Job der Liste die Markierung "2". Jeder seiner Folgejobs, der die Markierung "1" hat, erhält ebenfalls die Markierung "2". Jobs, deren Markierung ungleich "1" ist, werden übergangen. Hat man die Folgejobs überprüft, fährt man mit dem nächsten Job der Liste fort. Ist seine Markierung auf "2" gesetzt,

---

1) vgl. dazu S.87.

überprüft und ändert man gegebenenfalls die Markierungen seiner Folgejobs usw., bis man alle Jobs durchlaufen hat. Auch in diesem Fall wird die zweite Stufe des Verfahrens solange wiederholt, bis sich keine Markierungsänderungen mehr ergeben. Im Beispiel werden sofort bei Job 1 auch die Markierungen der Jobs 2 und 3 auf "2" gesetzt. Beim zweiten Durchlaufen der Jobs werden keine Markierungsänderungen mehr vorgenommen.

Die mit "2" gekennzeichneten Jobs bilden einen Zyklus und können entsprechend gekennzeichnet werden. Sie werden aus der Matrix entfernt (auch als Folgejobs). Mit der Restliste wird wieder bei der ersten Stufe begonnen, bis alle Zyklen gefunden worden sind und die Liste erschöpft ist. Tabelle 20 zeigt die beiden Stufen des Verfahrens zur Zyklussuche für das Beispiel aus Tabelle 18 noch einmal auf.

Tab. 20: Feststellung der Schleife für das Beispiel aus Tab. 18

Job-Nr.	Folgejobs	Markierungen				
		1. Stufe				2. Stufe
1	2,3	0	1	1	1	2
2	1,3	0	0	1	1	2
3	1,2	0	0	0	1	2

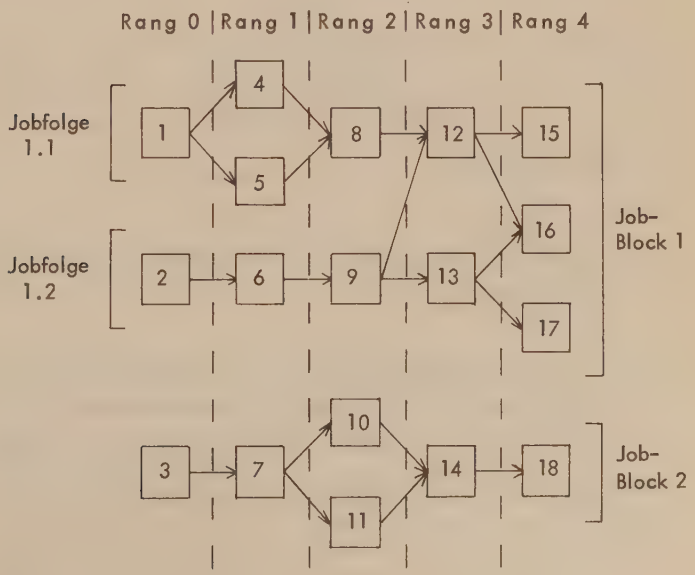
Allerdings kommt es bei diesem Verfahren vor, daß in einem großen Zyklus noch kleinere Schleifen enthalten sind, die nicht gesondert ausgewiesen werden. Wird bei einem Durchlauf nur ein Job ausgesondert, ist er selbstverständlich nicht als Zyklus anzusehen, sondern nur aus der Liste auszusondern. Führt man nach einer Auflösung der ausgedruckten Zyklen eine neue Rangrechnung durch, darf kein Rang mehr auftreten, der gleich der Anzahl der in der Matrix gespeicherten Jobs ist.

Der Arbeitsvorbereitung und Operation ist mit dieser Einteilung der Jobs in Rangklassen ein wertvolles Hilfsmittel an die Hand gegeben. Für die praktische Arbeit muß diese Einteilung aber noch weiter verfeinert werden. Die Jobs, die an einem Tag laufen müssen, gehören den verschiedensten

Arbeitsgebieten an. In den meisten Fällen sind diese Arbeitsgebiete sachlich völlig unabhängig voneinander, können also auch datenverarbeitungsmäßig ohne Rücksicht auf andere Sachgebiete erledigt werden. Diese Einteilung der Jobs in sachlich zusammenhängende sog. Jobblöcke kann im Anschluß an die Rangrechnung - eventuell nach erfolgter Zyklusauflösung - ebenfalls per Programm vorgenommen werden.

Dazu wird das Feld, das die Jobnamen, die Rangziffern und die Folgejobs enthält, aufsteigend nach Rängen sortiert. Dem ersten Job des Feldes ordnet man die Blocknummer 1 zu. Dann sucht man die Folgejobs des gekennzeichneten Jobs auf und gibt ihnen die gleiche Blocknummer. Genauso verfährt man mit den Folgejobs aller bereits gekennzeichneten Jobs. Da die Folgejobs eines Jobs im Rang i mindestens den Rang i + 1 besitzen, kann bei einmaligem Durchlaufen des Feldes immer eine zusammenhängende Jobfolge gekennzeichnet werden.

Abb. 18: Beispiel zur Job-Blockbildung für 18 Jobs 1 bis 18



Hat man eine Jobfolge gekennzeichnet, wählt man den nächsten Job im Rang 0 aus und gibt ihm die nächste freie Blocknummer. Wie bereits beschrieben, verfährt man auch mit seinen Folgejobs und deren Folgejobs. Dabei kommt es häufig vor, daß ein Folgejob bereits mit einer niedrigeren Blocknummer gekennzeichnet ist (s. Abb. 18: Jobs 12 und 16). Die bis zu diesem Job selbständigen Jobfolgen 1.1 und 1.2 vereinigen sich in diesen Punkten zu einer gemeinsamen Jobfolge, und beide Folgen gehören zu Jobblock 1. Stellt sich also heraus, daß der Nachfolger eines Jobs bereits eine niedrigere Blocknummer besitzt, so werden auch alle anderen Jobs, die bereits die höhere Nummer erhalten haben, mit der niedrigeren Blocknummer versehen. Auch für die übrigen Folgejobs gilt die niedrigere Nummer. Hat man die gesamte Jobfolge abgearbeitet, wählt man wieder den nächsten Job im Rang 0 aus usw.

Gibt es keinen Job im Rang 0 mehr, müssen bei diesem Vorgehen alle Jobs mit einer Blocknummer versehen sein. Ein Block kann dabei aus nur einem Job bestehen, er kann e i n e Jobfolge umfassen und er kann sich aus mehreren Jobfolgen zusammensetzen.

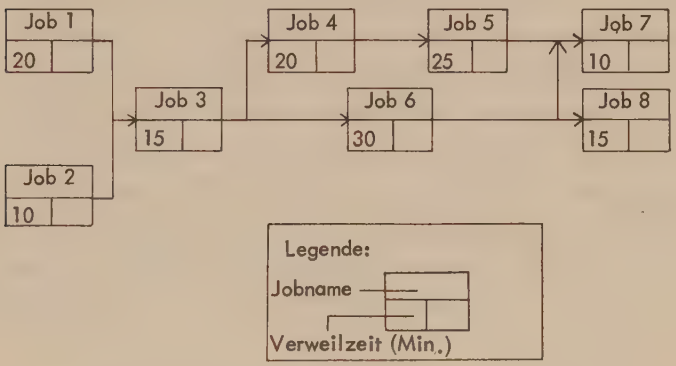
Ein letzter Schritt der Blockbildung kann noch darin bestehen, daß man die gefundenen Jobblöcke nach Abgabeterminen sortiert und entsprechend umbenennt, um auch die Reihenfolge für die Erledigung der Jobs in der Ablaufplanung festzulegen.

4.3 Die Terminplanung für den Jobablaufplan einer Planungsperiode

Aufbauend auf dem Jobablaufplan, der mit Hilfe eines der beiden in Kapitel 4.2 dargestellten Rangrechnungsverfahrens sowie dem Verfahren zur Bildung von Jobblöcken gewonnen wird, können mit der T e r m i n p l a n u n g weitere Ausgangsdaten für die Belegungsplanung eines Rechenzentrums be-

stimmt werden[1]. In der Terminplanung wird die früheste und späteste zeitliche Lage der einzelnen Jobs innerhalb eines Jobblocks festgelegt. Sie geht von den logischen Abhängigkeiten der Jobs sowie den für die einzelnen Jobs durch Beobachtungen über einen längeren Zeitraum festgestellten Verweilzeiten aus. An dem einfachen Beispiel der Abb. 19 soll die dem Terminplan zugrunde liegende Zeitrechnung[2] erläutert werden. Voraussetzung für ihre Durchführung ist ein zyklenfreier Jobstrukturplan. Weiterhin wird angenommen, daß ein Job während seiner Ausführung nicht unterbrochen werden darf.

Abb. 19 : Beispiel zur Zeitrechnung



Die Zeitrechnung zerfällt in die "Vorwärtsrechnung", bei der die frühesten Beginn- und Endzeitpunkte eines Jobs ermittelt werden, und in die sog. "Rückwärtsrechnung", bei der man die spätesten Anfangs- und Endzeitpunkte der Jobs in bezug auf die Ge-

1) Da die Zeitwerte in der Terminplanung ohne Berücksichtigung von Kapazitäten ermittelt werden, werden sie meist nicht mit den bei der Belegungsplanung vergebenen Werten übereinstimmen, die kapazitive Engpaßsituationen berücksichtigen.

2) Die Zeitrechnung soll hier nur kurz dargestellt werden. Eine ausführliche Erläuterung findet man u.a. in dem Begleitbuch zum Fernsehlehrgang "Netzplantechnik": AUTORENTEAM, Netzplantechnik - ein Fortbildungskurs im Medienverbund Fernsehen - Lehrbuch - Seminare, VDI-Verlag GmbH, Düsseldorf 1971, S. 17 ff.

samtdauer des entsprechenden Jobblocks bestimmt. Als Startzeitpunkt des Jobblocks wird der Zeitpunkt 0 angenommen, d.h. die Jobs 1 und 2 können, da sie keine Vorjobs besitzen, zu diesem Termin beginnen. Allgemein gelten für die Bestimmung der Frühesten Anfangs-Zeitpunkte und der Frühesten End-Zeitpunkte von Jobs - abgekürzt FAZ und FEZ - die beiden folgenden Beziehungen:

$$\text{Frühester Anfangs-Zeitpunkt} \quad FAZ_j = \max_{\text{alle } i} (FEZ_i)$$

$$\text{Frühester End-Zeitpunkt} \quad FEZ_j = FAZ_j + D_j \ ,$$

wobei i stets einen Vorjob (Vorjobs) von j bezeichnet. Bei der Berechnung des FAZ eines Jobs betrachtet man jeweils nur einen Ausschnitt des entsprechenden Jobblocks, nämlich den betrachteten Job und seine sämtlichen Vorjobs. Der FAZ eines Jobs kann ermittelt werden, wenn die frühesten End-Zeitpunkte FEZ der Vorjobs bekannt sind. Der größte FEZ eines unmittelbaren Vorjobs ist der FAZ des betrachteten Jobs. So kann beispielsweise der FAZ von Job 7 erst errechnet werden, wenn die FEZ der Jobs 5 und 6 feststehen. Den FEZ eines Jobs erhält man durch die Addition seines frühesten Anfangs-Zeitpunkts FAZ und seiner Verweilzeit. In der Vorwärtsrechnung wird der Jobblock so vom Startjob bis zum Zieljob durchlaufen.

In der Rückwärtsrechnung geht man davon aus, daß der früheste End-Zeitpunkt des (der) Zieljobs bzw. die gesamte Blockdauer nicht überschritten wird. Man ermittelt dann für jeden Job den Spätesten Anfangs-Zeitpunkt (SAZ), zu dem er spätestens begonnen werden muß, und den Spätesten End-Zeitpunkt (SEZ), zu dem er spätestens beendet sein muß, um die Gesamtdauer des Jobblocks einzuhalten. Dazu setzt man die spätesten Endtermine der Jobs, die keine Folgejobs besitzen, dem frühesten Endzeitpunkt des Jobs mit dem größten FEZ gleich und bestimmt die spätesten Termine ihrer Vorjobs nach den Beziehungen:

$$\text{Spätester Endzeitpunkt} \quad SEZ_i = \underset{\text{alle } j}{MIN}\,(SAZ_j)$$

$$\text{Spätester Anfangszeitpunkt} \quad SAZ_i = SEZ_i - D_i \;,$$

wobei j ein Folgejob von i ist. Während man bei der Vorwärtsrechnung den jeweils größten FEZ eines Vorjobs als FAZ eines Jobs festsetzt, ist bei der Rückwärtsrechnung folgerichtig der kleinste SAZ eines unmittelbaren Folgejobs bestimmend für den SEZ des betrachteten Jobs, denn der Job muß spätestens beendet sein, wenn der erste Folgejob starten muß. Kennt man den SEZ eines Jobs, erhält man seinen SAZ durch Subtraktion seiner Verweilzeit von seinem SEZ.

Aus den Terminen der einzelnen Jobs kann außerdem eine Hilfsgröße berechnet werden, die man als gesamte Pufferzeit (GP) [1] bezeichnet. Die gesamte Pufferzeit gibt an, um welche Zeitspanne man einen Job verschieben kann, ohne den Endtermin des Jobblocks hinauszuschieben. Bei der Berechnung der gesamten Pufferzeit wird angenommen, daß sich der betrachtete Job in seiner frühesten Lage befindet und daß seine Folgejobs ihre späteste Lage einnehmen. Daraus ergibt sich, daß man zur Ermittlung der Pufferzeit die Differenz der End- bzw. Anfangs-Zeitpunkte zu bilden hat:

$$GP_i = SEZ_i - FEZ_i = SAZ_i - FAZ_i .$$

Tab. 21 zeigt die Ergebnisse der Zeitrechnung für den Jobblock aus Abb. 19.

---

1) Es handelt sich bei dem Ausdruck "gesamte Pufferzeit" um einen genormten Ausdruck aus dem Bereich der Netzplantechnik (DIN-Norm 69900).

Tab. 21 : Ergebnisse der Zeitrechnung des Beispiels aus Abb. 19 in Minuten

Jobname	Verweilzeit	Frühester Anfangs-Zeitpunkt FAZ	Frühester End-Zeitpunkt FEZ	Spätester Anfangs-Zeitpunkt SAZ	Spätester End-Zeitpunkt SEZ	Gesamte Pufferzeit GP
1	20	0	20	0	20	0
2	10	0	10	10	20	10
3	15	20	35	20	35	0
4	20	35	55	35	55	0
5	25	55	80	55	80	0
6	30	35	65	45	75	10
7	10	80	90	80	90	0
8	15	65	80	75	90	10

Die Jobs 1 und 2 erhalten in der Vorwärtsrechnung als FAZ den Wert 0 zugewiesen, da sie keine Vorjobs besitzen. Job 1 ist nach 20 Minuten erledigt, also erhält er den FEZ 20, Job 2 entsprechend den FEZ 10. Job 3 ist vom Ende der Jobs 1 und 2 abhängig. Ihm wird als FAZ der größte FEZ eines seiner Vorjobs zugeordnet, in diesem Fall also der FEZ von Job 1 (Zeitpunkt 20); der FEZ von Job 3 ist 35. Diesen FEZ erhalten die Jobs 4 und 6 als FAZ, usw. Den größten FEZ nach der Vorwärtsrechnung hat Job 7 mit einem FEZ von 90 Minuten und das bedeutet, daß der gesamte Jobblock in einer Gesamtverarbeitungszeit von 90 Minuten erledigt werden kann.

In der Rückwärtsrechnung werden die Jobs in der Gegenrichtung durchlaufen, wobei den Jobs 7 und 8 der SEZ 90 (Blockdauer) zugeordnet wird. Der SAZ von Job 7 ist dann der Zeitpunkt 80, der SAZ von Job 8 der Zeitpunkt 75. Der SAZ von Job 7 wird als SEZ für Job 5 übernommen, da nur Job 7 Folgejob von Job 5 ist. Als SEZ von Job 6 wird der kleinere der spätesten Anfangs-Zeitpunkte der Jobs 7 und 8 gewählt, da beide Folgejobs von Job 6 sind. Der SEZ von Job 6 ist demnach der Zeitpunkt 75 und sein SAZ der Zeitpunkt 45. Die übrigen Zeitpunkte der Rückwärtsrechnung werden entsprechend ermittelt.

Nun stellen die Ergebnisse dieser Zeitrechnung aber noch nicht den eigentlichen Terminplan dar, denn dieser entsteht erst, wenn man die absoluten Zeitpunkte durch die vom Beginn des Planungszeitraums an geltende Uhrzeit ersetzt. Tab. 22 zeigt den Terminplan, der von dem Startzeitpunkt 14.00 Uhr ausgeht:

Tab. 22 : Terminplan für den Jobblock aus Abb. 19

Jobname	Verweilzeit	Frühester Anfangs-Termin FAT	Frühester End-Termin FET	Spätester Anfangs-Termin SAT	Spätester End-Termin SET	Gesamt Puffer GP
1	20	14.00	14.20	14.00	14.20	0
2	10	14.00	14.10	14.10	14.20	10
3	15	14.20	14.35	14.20	14.35	0
4	20	14.35	14.55	14.35	14.55	0
5	25	14.55	15.20	14.55	15.20	0
6	30	14.35	15.05	14.45	15.15	10
7	10	15.20	15.30	15.20	15.30	0
8	15	15.05	15.20	15.15	15.30	10

Eine eingehende Erläuterung dieses Terminplans erübrigt sich, da die Zeitpunktrechnung aus Tab. 21 in Tab. 22 nur in Termine umgesetzt worden ist, die für einen bestimmten Planungszeitraum Gültigkeit besitzen.

Führt man die Terminplanung bei allen Jobblöcken durch, so bietet sie den Vorteil, daß man in jedem Jobblock den sog. "kritischen Weg" kennenlernt. Auf dem kritischen Weg liegen die Jobs, deren gesamte Pufferzeit gleich 0 ist. Die terminliche Verschiebung eines dieser Jobs würde - wie leicht einzusehen ist - zu einer Erhöhung der Blockdauer führen. Jobs, deren gesamte Pufferzeit größer als 0 ist, können dagegen im Rahmen ihrer Pufferzeit ohne Einfluß auf die Blockdauer verschoben werden[1].

1) Dabei ist darauf zu achten, inwieweit die Verschiebung eines Jobs sich auf die gesamte Pufferzeit seiner Folgejobs auswirkt.

4.4 Beschreibung eines Programms zur Job-Ablaufplanung

Die Ablaufplanung kann in jedem Rechenzentrum ohne große manuelle Eingriffe mit Hilfe eines Programms durchgeführt werden, das die Auswahl der Jobs für eine Planungsperiode, die Feststellung der Abhängigkeiten, die Rangrechnung und Zyklussuche, die Blockbildung und gegebenenfalls die Terminplanung vornimmt. In diesem Abschnitt 4.4 soll deshalb ein vom Verfasser in FORTRAN IV geschriebenes Programm beschrieben werden, mit dessen Hilfe täglich die Planung für eine IBM/370-165 und eine IBM/370-155 durchgeführt wird [1]. Über eine Parameterkarte wird dabei gesteuert, ob

- ein Gesamtplan für beide Anlagen,
- zwei Einzelpläne oder
- ein Plan für eine der Anlagen

ausgegeben werden soll. Durch die Angabe bestimmter weiterer Parameter ist zusätzlich die Ausgabe von Ablaufplänen

- für einen beliebig festlegbaren Zeitraum
- für verschiedene Firmen bzw. Tochtergesellschaften und
- für Jobnummerintervalle

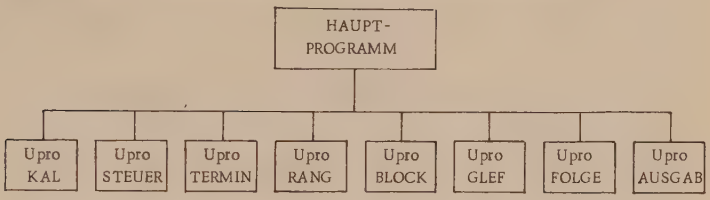
möglich.

Das Programm besteht aus einem Haupt- und acht Unterprogrammen. Es benötigt - vor allem auf Grund der großen Tabellen, die für die verschiedenen Berechnungen und Vergleiche notwendig sind - insgesamt 150 K-Bytes. Durch Anwendung der sog. "Überlagerungstechnik", bei der neben dem Hauptprogramm nur das (die) gerade durchzuführende(n) Unterprogramm(e) im Kernspeicher steht (stehen), kann die Programmgröße auf etwa 90 K-Bytes verringert werden. Die Programmstruktur ist bei dem vorliegenden Programm sehr einfach, da die Unterprogramme nur jeweils vom Hauptprogramm aufgerufen werden:

---

1) Das Programm nimmt keine Terminplanung vor.

Abb. 20 : Programmstruktur des Ablaufplanungsprogramms



Die Aufgabe des Hauptprogramms besteht vor allem darin, die Folgejobs der einzelnen Jobs durch Datei-Vergleiche festzustellen und zu speichern. Alle übrigen Berechnungen und die Ausgabe der Planungsliste geschehen in Unterprogrammen . Dazu wird im Unterprogramm KAL einmal im Jahr ein Kalender aufgebaut, der Arbeits- und Feiertage bzw. Nichtarbeitstage enthält. Das Unterprogramm STEUER erstellt mit Hilfe dieses Kalenders und des Maschinendatums das für den Planungstag gültige Terminsteuerwort, das dann in TERMIN mit den Terminsteuerworten der einzelnen Jobs der Jobstammdatei verglichen wird. Die Jobstammsätze der Jobs, die am Planungstag laufen müssen, werden in eine für diesen Tag gültige Planungsdatei geschrieben. Häufig genügt es nicht, nur die terminlich durch die Terminsteuerworte fixierten Jobs auszuwählen, da es neben den Jobs, die zumindest im monatlichen Rhythmus im voraus terminlich festzulegen sind, in jedem Rechenzentrum einige Jobs gibt, die nur auf Abruf der entsprechenden Fachabteilung laufen. Diese Jobs werden durch sog. "Abrufkarten" ebenfalls aus der Jobstammdatei in die Planungsdatei übernommen. Entsprechend kann durch Löschkarten die Speicherung von Jobs in der Planungsdatei verhindert werden, die trotz der terminlichen Fixierung auf den Planungstag noch nicht laufen sollen.

Nach der Feststellung der Folgejobs wird im Unterprogramm RANG die Rangrechnung und Zyklussuche durchgeführt. Stellt sich dabei heraus, daß Zyklen vorliegen, bricht das Programm an dieser Stelle ab, nachdem es alle aufgetretenen Zyklen ausgedruckt hat. Diese Zyklen werden von der Sachbearbeitung oder der Arbeitsvorbereitung durch Angabe der gültigen Reihenfolge aufgelöst, und die entsprechenden Jobs werden über sog. "erweiterte Abrufkarten" neu in das Programm eingegeben. Die

Auflösung geschieht, indem die gültigen Folgejobs für jeden Job des Zyklus vorgegeben werden. Man hat damit insgesamt folgende drei Möglichkeiten, Folgejobs festzulegen:

1. durch eine Festlegung im Feld "Folgejobs der Jobstammdatei" bei ständig auftretenden Zyklen,
2. durch eine Angabe in den "erweiterten Abrufkarten" bei wechselnden Zyklen oder
3. durch den Vergleich der Ein- und Ausgabedateien im Programm selbst.

Während bei einer Festlegung der Abhängigkeiten durch erweiterte Abrufkarten oder der Angabe von Folgejobs die Dateistruktur nicht untersucht wird, erfolgt die dritte Festlegungsart automatisch vom Programm, wenn keine der beiden ersten Formen gewählt wurde.

Da die Jobfolgen auf Grund der Dateiabhängigkeiten zwischen den Jobs festgestellt werden, ist es notwendig, daß die Datei-Informationen, die für jeden Job in der Jobstammdatei enthalten sind, den tatsächlich angezogenen Files entsprechen. Es muß deshalb zusätzlich eine Möglichkeit geschaffen werden, einem Job für einen ganz bestimmten Lauf andere Files zuzuordnen, als er sie im Normalfall benutzt. Dies geschieht durch sog. "Fileänderungskarten" , die man zusätzlich zu einer Abrufkarte eingibt. Diese Karten erlauben es, neue Dateien zu definieren, bestehende Dateien zu verändern (z.B. durch Angabe einer älteren Generationsnummer) oder bestehende Dateien für den Planungstag zu löschen. Der Vorteil dieser nur vorübergehend zu berücksichtigenden Informationen besteht darin, daß die entsprechenden Dateien nur in der Planungs- und nicht in der Jobstammdatei berücksichtigt werden.

Das Unterprogramm BLOCK führt für die in der Planung stehenden Jobs die Blockbildung durch. Durch einen zusätzlichen Sortiervorgang kann ohne Schwierigkeiten die Aufeinanderfolge der Blöcke so festgelegt werden, daß sie nach Abgabeterminen aufsteigend geordnet sind.

Nach der Durchführung der Blockbildung sind die Probleme der Job-Ablaufplanung im Grunde gelöst. Solange jedoch die Ausführung der Jobs vom Maschinenpersonal vorgenommen wird, ist es zweckmäßig, noch weitere Ablauf-Informationen vom Programm ausgeben zu lassen. Aus speicherungstechnischen Gründen befinden sich in den meisten Rechenzentren zur Zeit noch Dateien, die Stammdaten enthalten, auf Magnetbändern. Diese sog. Stammbänder werden in vielen Jobs benötigt, stehen aus technischen Gründen aber immer nur einem Job während seiner Laufzeit zur Verfügung. Die Information, welche Jobs gleiche Eingabedateien erfordern, ist für die Operation daher sehr wichtig, denn diese Jobs dürfen im Multiprogramming nicht parallel zueinander abgefahren werden. Das Unterprogramm GLEF (Untersuchung auf gleiche Eingabefiles) ermittelt die Jobs mit gleichen Eingabedateien und gibt diese Zusatzinformation an das Ausgabe-Unterprogramm. Dabei ist bei der Bestimmung von Jobs mit gleichen Eingabedateien zu beachten, daß sie nicht parallel laufen dürfen, wenn sie innerhalb eines Block im gleichen Rang gleiche Eingabedateien benutzen, wenn sie innerhalb eines Blocks in verschiedenen Rängen, aber unterschiedlichen Jobfolgen gleiche Dateien verlangen oder wenn sie in verschiedenen Blöcken die gleichen Dateien anziehen.

Als zusätzliche Information werden im Unterprogramm FOLGE schließlich noch die Jobnamen der Folgejobs aus den internen Jobnamen[1] gewonnen, mit denen alle Berechnungen durchgeführt wurden. In der praktischen Arbeit hat sich die Angabe dieser Jobnamen vor allem bei umfangreichen Blöcken mit mehreren Jobfolgen sehr gut bewährt.

Das Unterprogramm AUSGAB dient der Erstellung der Ablaufplanungslisten. Der Listenaufbau wird sich immer an den Anforderungen des jeweiligen Rechenzentrums orientieren müssen, für das die Ablaufplanung durchgeführt wird. Der in Anhang 2 (S. 221) gezeigten Aufbau kann daher nur als ein Beispiel unter vielen möglichen Formen angesehen werden[2].

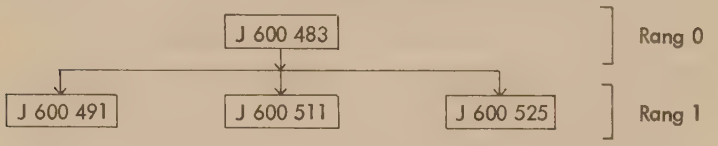
---

1) vgl. dazu S. 86.

2) Die Erläuterung der einzelnen Informationen kann der Legende in Anhang 2 entnommen werden.

Anhang 2 zeigt die logischen Abhängigkeiten des 19. festgestellten Jobblocks auf, der aus 4 Jobs besteht. Die Jobstruktur dieses Blocks gibt Abb. 21 wieder.

Abb. 21 : Jobstruktur des in Anhang 2 gezeigten Jobblocks



Interessant ist bei dem in Anhang 2 abgebildeten Block, daß die drei Jobs, die im Rang 1 stehen, nicht parallel zueinander laufen dürfen, weil sie gleiche Eingabedatenbestände benötigen[1]. Die Reihenfolge ihrer Erledigung spielt dabei allerdings keine Rolle und kann in Abhängikeit von den jeweils zur Verfügung stehenden Kapazitäten erfolgen. Pro Jobblock werden in dieser vom Verfasser vorgeschlagenen Ausgabeliste eine oder gegebenenfalls auch mehrere Seiten ausgegeben, die es dem Bedienungspersonal ermöglichen, auf Grund der Kernspeicher-, Bandeinheiten- und Plattenspeicheranforderungen sowie der vorgegebenen Abhängigkeiten die Anlage möglichst gut auszulasten.

1) vgl. dazu in Anhang 2 bei Job J 600491 die Meldung: "Gleicher E-File in: J 600511, J 600525" und die entsprechenden Meldungen bei diesen Jobs.

5 DARSTELLUNG DER KAPZITÄTSAUSGLEICHSVERFAHREN

5.1 Definition der Zielvorstellung für die Belegungsplanung und Diskussion verschiedener Kapazitätsplanungskriterien

Die in Abschnitt 4.3 beschriebene Terminplanung für die Jobblöcke eines Planungszeitraums kann der Arbeitsvorbereitung Hinweise für die Belegungsplanung eines Rechners geben. Wird unterstellt, daß die zu jedem Zeitpunkt benötigten Maschinenkapazitäten in ausreichendem Umfang zur Verfügung stehen, könnten die Jobs bereits auf Grund dieses Terminplans erledigt werden. Daß es sich hierbei allerdings nicht um eine realistische Annahme handelt, erkennt man, wenn die Inanspruchnahme der Kapazitäten durch die zu verarbeitenden Jobs mit der Kapazitätsverfügbarkeit verglichen wird.

Eine vom Verfasser durchgeführte Untersuchung der Jobablaufpläne eines Monats[1)] ergab, daß bei insgesamt 150 - 200 Jobs pro Tag etwa 40 - 50 voneinander unabhängige Jobblöcke auftraten. Tab. 23 zeigt z.B. die Anzahl der Jobs pro Jobblock bei 45 Jobblöcken auf.

1) Diese Untersuchung wurde für die ca. 1800 Routinejobs eines Großrechenzentrums in der Stahlindustrie im Oktober 1972 durchgeführt; sie bezog sich auf die Jobablaufpläne für eine Anlage des Typs IBM/370-165, deren Konfiguration in Kapitel 6.2 näher erläutert wird.

Tab. 23 : Verteilung von 45 Jobblöcken nach der Anzahl der Jobs pro Jobblock

Anzahl Jobs pro Jobblock	Jobblöcke	
	Anzahl	Anteil
1	15	33,33 %
2	12	26,67 %
3 - 5	10	22,22 %
6 - 10	5	11,11 %
> 10	3	6,67 %
Σ	45	100,0 %

Der Anteil der Jobblöcke mit 6 und mehr Jobs machte bei der Untersuchung nur 17,78 % aus. Bei den größeren Jobblöcken überwog die Zahl der Jobs in den Rängen 0, 1 und 2 bei weitem. Die manuell durchgeführte Zeitrechnung ergab für die einzelnen Blöcke Verarbeitungszeiten, die zwischen 10 Minuten und 600 Minuten lagen; die durchschnittliche Verarbeitungszeit betrug je Block 85 Minuten. Vernachlässigt man einmal die Tatsache, daß etwa 15 - 20 % der Jobblöcke Bereitstellungs- und/oder Abgabetermine [1] besitzen, die ihre Erledigung erst zu einem späteren Zeitpunkt erlauben, zeigt sich, daß zu Beginn jeder Planungsperiode ein sehr großer Kapazitätsbedarf besteht, während am Ende des Planungszeitraums die Maschinenkapazitäten nicht ausgelastet sind. Natürlich kann die Konfigurierung einer Datenverarbeitungsanlage nicht am Maximalbedarf ausgerichtet werden, der sich auf Grund des Terminplans zu Beginn jeder Planungsperiode ergibt. Vielmehr wird die Anlage so ausgelegt werden, daß im Verlauf des gesamten Planungszeitraums die anstehenden Arbeiten erledigt werden können. Daraus ergibt sich die Notwendigkeit, die Lauftermine von Jobs und Jobblöcken zu verschieben, und zwar selbstverständlich unter Berücksichtigung der Bereitstellungs- und Abgabetermine einzelner Jobs. Die Verlagerung von Jobblöcken bei gleichzeitig möglichst optimaler Kombination der Jobs im Hin-

1) vgl. hierzu S. 58 dieser Arbeit.

blick auf die zur Verfügung stehende Kapazität ist Aufgabe der Belegungsplanung. Durch ein geeignetes Verfahren können dabei durch die zeitliche Verlagerung der Jobs Kapazitätsüber- bzw. -unterschreitungen vermieden, kann ein an der Kapazität der Anlage ausgerichteter Terminplan aufgestellt und kann bei Beobachtungen über einen längeren Zeitraum hinweg erkannt werden, ob die Konfiguration der Datenverarbeitungsanlage den Anforderungen genügt oder ob einzelne Systemkapazitäten in zu großer bzw. zu kleiner Anzahl zur Verfügung stehen.

Das mit der zeitlichen Verlagerung der Jobs auftretende Problem gehört zu dem großen Kreis der Reihenfolge- und Zuordnungsprobleme. Schon das in Kapitel 4 behandelte Problem der automatischen Festlegung von Jobstrukturen konnte mit einer gewissen Berechtigung als Reihenfolgeproblem bezeichnet werden. Erst durch die Zuordnung nur beschränkt zur Verfügung stehender Kapazitäten unter optimierenden Gesichtspunkten und die daraus resultierende genaue, zeitliche Festlegung der Jobs auf bestimmte Termine wird die Belegungsplanung jedoch zu einem Reihenfolgeproblem im engeren Sinn. STAHLKNECHT [1] umreißt die in diesem Zusammenhang auftretenden Fragen wie folgt: "Reihenfolgeprobleme sind Optimierungsprobleme. Sie bestehen aus einer Anzahl von Entscheidungsmöglichkeiten und aus einer Zielfunktion, deren Wert von der Reihenfolge abhängt, in der diese Entscheidungen vorgenommen werden."

Bei der Definition der Zielvorstellung, die in dieser Arbeit mit der Belegungsplanung verfolgt wird, sind zwei voneinander unabhängige Problemkreise zu trennen. Es geht erstens um die Festlegung des eigentlichen Optimierungsziels, d.h. um die Frage, welche Zielgröße im Rahmen der Belegungsplanung optimiert werden soll, und es ist zweitens die Frage zu klären, für welchen Zeitraum und mit Hilfe welcher Methode die Belegungsplanung durchgeführt werden soll. Als Planungszeitraum für die Jobablaufplanung bietet sich der Arbeitstag an, weil die Jobabhängigkeiten nur für

---

1) P. STAHLKNECHT, Operations Research, 2. Aufl., Braunschweig 1970, S. 197.

einen vollen Tag sinnvoll bestimmt werden können. Für eine Belegungsplanung, die die genauen Lauftermine der Jobs vorgeben muß, wird dagegen zu prüfen sein, ob nicht ein kürzeres Planungsintervall zu günstigeren Ergebnissen führt. Die Bedeutung der Frage nach der Methode der Belegungsplanung wird klar, wenn man sich die bei der Darstellung der Betriebssysteme gemachten Ausführungen in das Gedächtnis zurückruft. Dort war deutlich geworden, welche Möglichkeiten der Anwender hat, den Jobablauf zu beeinflussen. Sobald das Betriebssystem die Kontrolle über die Jobs übernimmt, bestehen für ihn keine Steuerungsmöglichkeiten mehr. Die Belegungsplanung muß also in dem Freiraum vorgenommen werden, der dem Benutzer vom Betriebssystem gewährt wird, denn es kann von der Anwendungsseite her gesehen nicht das Ziel sein, in Routinen des Betriebssystems einzugreifen. Es stellt sich vielmehr die Aufgabe, mit der Belegungsplanung Ziele zu verfolgen, die die Einplanungsverfahren der Betriebssysteme ergänzen.

Bevor diese Optimierungsziele im einzelnen festgelegt werden, ist es jedoch notwendig, den Begriff der Kapazität einer Datenverarbeitungsanlage zu definieren. In der Betriebswirtschaftslehre versteht man unter Kapazität das "Leistungsvermögen eines Betriebsmittels je Zeiteinheit" [1]. Diese Definition besitzt - übertragen auf Datenverarbeitungsanlagen - wenig Aussagekraft, d.h. sie muß für diesen Bereich konkretisiert werden. Die Bestimmung der Kapazität einer EDV-Anlage kann sowohl für das Gesamtsystem als auch für die einzelnen Systemelemente vorgenommen werden. Die meisten Größen, die bisher in der Literatur vorgeschlagen wurden, beschreiben das Leistungsvermögen der Gesamtanlage, um den Vergleich verschiedener Datenverarbeitungsanlagen zu ermöglichen. Dazu werden einzelne Größen wie Additionszeiten der Zentraleinheit, Kernspeicherzykluszeiten und einfache Programme mit repräsentativen Befehlszusammenstellungen, aber auch Systeme von Leistungsparametern definiert, die das Gesamtverhalten eines im Multiprogramming

---

1) G. WÖHE, Einführung in die Allgemeine Betriebswirtschaftslehre, 6. Aufl., Berlin 1964, S. 80.

betriebenen Systems erfassen[1], wie z.B. die Druchsatzrate von Jobs im Multiprogramming und der zeitliche Gewinnfaktor des Multiprogramming gegenüber der sequentiellen Verarbeitungsform.

Die optimierende Belegungsplanung ist nicht möglich, wenn die Höhe der von den einzelnen Systemelementen einer Datenverarbeitungsanlage zur Verfügung gestellte Kapazität unbekannt ist, und aus diesem Grund muß die Gesamtkapazität einer Datenverarbeitungsanlage in einzelne Systemkapazitäten unterteilt werden. Solche Systemkapazitäten (system resources) sind die Zentraleinheit, der Kernspeicher, Ein-/Ausgabeeinheiten, Speichermedien und Ein-/Ausgabekanäle. Alle diese Systemelemente stehen dem Benutzer in einer vorgegebenen Anzahl zur Verfügung, sind jedoch nicht alle in gleichem Umfang durch den Anwender beeinflußbar. Für die Belegungsplanung muß deshalb untersucht werden, bei welchen Systemelementen eine Steuerung durch den Benutzer möglich ist.

Der Kontrolle des Benutzers vollständig entzogen ist die Zuteilung der Zentraleinheit sowie die Vergabe der Ein-/Ausgabekanäle an die einzelnen Tasks. Es kommt hinzu, daß diese Systemelemente den einzelnen Aufgaben nur für Sekundenbruchteile zugeordnet werden und allein auf Grund dieser Tatsache nicht einplanbar sind. Eine Beeinflussungsmöglichkeit ist hier nur insoweit gegeben, als man versuchen kann, bei der Belegungsplanung immer rechenintensive mit ein-/ausgabeintensiven Jobs zu kombinieren.

Anders steht es dagegen mit den übrigen Systemelementen, wie z.B. dem Kernspeicher, den Magnetband- und Magnetplattenspeichereinheiten. Sie werden über einen längeren Zeitraum von den einzelnen Tasks belegt und erlauben daher eine Einplanung durch den Benutzer, obwohl es natürlich auch hier nicht möglich ist, einem Job eine ganz bestimmte Einheit oder einen bestimmten Speicherbereich bei der Belegungsplanung zuzuteilen. Dafür be-

---

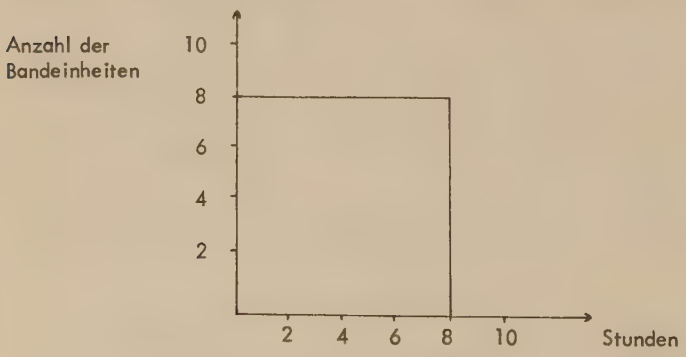
1) vgl. dazu u.a. K. KÜMMERLE, Charakteristische Größen zur Beschreibung der Leistungsfähigkeit und Effektivität von EDV-Anlagen, Elektronische Rechenanlagen, 1972, Heft 1, S. 12 - 18.

steht aber auch keine Notwendigkeit, da die einzelnen Einheiten und Speicher als technisch gleichwertig anzusehen sind. Unterschiedliche Systemelemente müssen aber natürlich getrennt eingeplant werden.

Als Maß für die Kapazität der einzelnen Systemkapazitäten kann der Flächeninhalt definiert werden, der sich aus der Multiplikation ihrer Anzahl mit der zur Verfügung stehenden Zeit ergibt.

Verfügt ein Rechenzentrum z.B. über 8 Bandeinheiten und arbeitet es maximal 8 Stunden pro Tag, so hat es eine Tageskapazität von 8 Bandeinheiten x 8 Stunden, d.h. von 64 "Bandeinheiten-Stunden". Dieser Flächeninhalt stellt die für die Belegungsplanung nutzbare Kapazität dar (vgl. Abb. 22). Auch für die anderen Systemelemente kann eine entsprechende Kapazitätsfestlegung vorgenommen werden.

Abb. 22: Verfügbare Kapazität von acht Bandeinheiten für einen Zeitraum von acht Stunden



Diese Art der Kapazitätsmessung erscheint recht grob. Eine feinere Messung, die z.B. an technischen Leistungsfaktoren, wie der Lese-/Schreibgeschwindigkeit von Magnetband- und Magnetplatteneinheiten ausgerichtet wäre, ist im Rahmen der Belegungsplanung jedoch nicht möglich, da die Nutzung der technischen Leistung vor allem programmabhängig ist (günstigste Satz- und Blocklängenfestlegung). Dieser Aspekt der Kapazität kann nicht

berücksichtigt werden, denn der Ausbildungsstand des Programmierungspersonals, der hierbei direkt mit angesprochen ist, muß kurzfristig als gegeben angenommen werden.

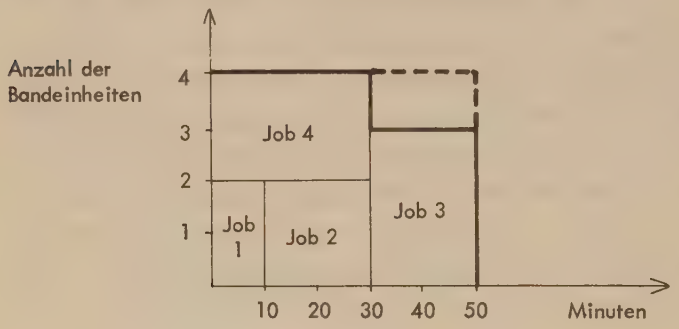
Abschließend zu diesen mit dem Begriff der Kapazität zusammenhängenden Fragen muß noch die K a p a z i t ä t s a u s l a s t u n g betrachtet werden. Die Betriebssysteme belegen die Systemkapazitäten stets nur für die augenblicklich zu erledigende Task, vom Benutzer aus gesehen also für einen Step. Genau genommen muß also auch die Messung der Kapazitätsauslastung stepbezogen vorgenommen werden, wenn exakte Aussagen über die Kapazitätsausnutzung gemacht werden sollen. Andererseits sollte die Belegungsplanung jobbezogen erfolgen, da der Anwender nicht den einzelnen Jobstep, sondern den gesamten Job als zu erledigende Arbeitseinheit betrachtet. An dieser Stelle muß eine E n t s c h e i d u n g getroffen werden, ob die Kenntnis der genauen Kapazitätsausnutzung, verbunden mit einer stepbezogenen Belegungsplanung, höher einzustufen ist als eine Jobplanung, die häufig eine zu hohe Kapazitätsausnutzung aufweisen kann, da sie sich an den maximalen Kapazitätsanforderungen orientieren muß, die eventuell nur ein Jobstep eines Jobs benötigt. Der zweiten Version ist aus praktischen Erwägungen heraus eindeutig der Vorzug zu geben, und zwar um so mehr, als eine Jobplanung auch rein zeitlich gesehen wesentlich schneller durchzuführen ist als eine stepbezogene Belegungsplanung.

Eine eigene Untersuchung für ein Großrechenzentrum hat außerdem ergeben, daß die Abweichungen von der maximal in Anspruch genommenen Kapazität in den einzelnen Jobsteps nur relativ gering sind.

Die nutzbare Kapazität für ein Systemelement wurde als Produkt aus der Anzahl der verfügbaren Einheiten dieses Elements und des Zeitraums definiert, in dem es für die Belegungsplanung zur Verfügung steht (vgl. Abb. 22). Es bietet sich an, die Messung der in Anspruch genommenen Kapazität analog hierzu durch Multiplikation der benutzten Bandeinheiten, Regions und Plattenspeicherbereiche mit den Verweilzeiten der ausgeführten Jobs vorzunehmen.

Trägt man die angeforderten Kapazitäten für jeden Kapazitätstyp im Zeitablauf in ein Koordinatensystem ein (s. Abb. 23), läßt sich die Belegung der Systemelemente aus dieser Grafik unschwer ablesen. Auch rechnerisch ist die Bestimmung des Kapazitätsausnutzungsgrads je Kapazitätstyp als Quotient aus der in Anspruch genommenen und der verfügbaren Kapazität ohne Schwierigkeiten durchführbar.

Abb. 23 : Beispiel einer Bandeinheitenbelegung für 4 Jobs



verfügbare Kapazität

benutzte Kapazität

Nach Klärung dieser Begriffe kann nun die bereits zu Beginn dieses Abschnitts aufgeworfene Frage nach dem Optimierungsziel geklärt werden. Die Optimierung des Einsatzes einer Datenverarbeitungsanlage führt bei einer kurz-, mittel- bzw. langfristigen Betrachtung zu unterschiedlichen Zielgrößen, die einer Beeinflussung unterliegen. Bei der kurzfristigen Optimierung , die einen Zeitraum bis zu etwa 3 Monaten umfassen sollte, ist nur die Zeit, in der die Anlage laufen muß, als Aktionsparameter anzusehen, d.h. die Systemkapazitäten werden kurzfristig als unveränderlich angenommen. Bei der mittelfristigen Optimierung können dagegen zusätzlich kapazitive Änderungen in der Systemkonfiguration in Betracht gezogen werden, da hier an einen Zeitraum von

3 Monaten bis zu einem Jahr gedacht ist, und bei der langfristigen Optimierung bezieht sich der Aktionsspielraum auf die Faktoren Zeit, Kapazität und Anlagentyp, d.h. langfristig muß auch ein Vergleich zwischen mehreren Datenverarbeitungssystemen vorgenommen werden.

Die Belegungsplanung stellt ab auf die Planungsperiode der Jobablaufplanung, d.h. in der Regel auf einen Arbeitstag, und daraus folgt, daß sie von einem gegebenen Datenverarbeitungssystem ausgehen muß. Ziel der Belegungsplanung kann also nur die kurzfristige Optimierung sein. Zwei einander gleichwertige Ziele lassen sich für die kurzfristige Betrachtung, d.h. bei vorgegebenem Maschinenpark, formulieren, nämlich die Maximierung des Jobdurchsatzes bei vorgegebner Gesamtverarbeitungszeit oder die Minimierung der Gesamtverarbeitungszeit bei vorgegebener Anzahl von Jobs.

Welches dieser beiden Ziele im konkreten Fall verfolgt werden soll, hängt von der Situation des einzelnen Rechenzentrums ab. Sind die Anlagen stark belastet, kann bei Verfolgung des ersten Ziels mit Hilfe der Belegungsplanung die Anlage eventuell noch für einen längeren Zeitraum in der bestehenden Konfiguration genutzt werden, bevor eine zeitliche Kapazitätsanpassung durch beispielsweise die Einführung einer zweiten, dritten oder vierten Schicht notwendig wird bzw. sie durch eine leistungsstärkere Anlage ersetzt wird. Ist die Konfiguration bereits im Hinblick auf zukünftige Aufgaben gekauft bzw. angemietet worden, ist sie also im Augenblick noch nicht voll ausgelastet, wird man vor allem die zweite Zielvorstellung verfolgen, um durch eine möglichst kleine Gesamtverarbeitungszeit Mietkosten einzusparen, die in einem direkten Zusammenhang mit der Anschaltzeit der Anlage stehen.

Durch die Verfolgung beider Ziele wird eine Kostenminimierung angestrebt, ohne daß in dieser Arbeit im einzelnen untersucht werden soll, wie sich der Kostenverlauf bei steigender oder sinkender Kapazitätsauslastung verändert[1].

Dagegen steht die Lösung des Problems noch aus, wie häufig die Belegungsplanung während einer Planungsperiode ausgeführt werden soll. Der Rahmen, der für die Beantwortung dieser Frage bleibt, läßt sich durch zwei extreme Planungsverfahren abstecken, nämlich 1. einer simultanen Belegungsplanung, die in kurzen Zeitabständen prüft, ob ein neuer Job gestartet werden kann, und 2. einer Belegungsplanung, die nur einmal für die gesamte Planungsperiode durchgeführt wird. Beide Verfahren sollen im folgenden näher erläutert werden.

Bei der Darstellung des SIEMENS-Betriebssystems wurde bereits eingehend das Zusatzprogramm DAB 3 ("Dynamische Anlagen Belegung") erörtert[2]. Es ist kennzeichnend für das erste Verfahren, bei dem das Optimierungsprogramm ergänzend zum Betriebssystem ständig mitläuft und entweder nach jedem Jobende oder nach einer bestimmten Zeitspanne aktiv wird und aus dem Bestand der einplanbaren Jobs mit Hilfe eines noch näher zu erläuternden Verfahrens den nächsten Job aussucht, der vom System erledigt werden soll. Dieses Vorgehen könnte man als simultane Belegungsplanung bezeichnen. Der große Vorteil dieser Planungsart ist darin zu sehen, daß das Problem der Abschätzung von Jobverweilzeiten in Multiprogramming-Systemen umgangen wird. Außerdem werden bei Jobabbrüchen die jeweiligen Folgejobs nicht freigegeben. Der Hauptnachteil des Verfahrens ist darin zu sehen, daß es ständig eine Kernspeicher-Region belegt. Bei der Komplexität des Planungsverfahrens, das alle aus dem Jobablaufplan bekannten Informationen verwerten muß, beträgt die Programmgröße nach

---

1) Die meisten EDV-Anlagen sind von den Benutzern nur angemietet und Kostenuntersuchungen könnten wegen der Vielfalt der häufig individuell geschlossenen Verträge nur an einem konkreten Unternehmen durchgeführt werden, was aus Geheimhaltungsgründen jedoch leider nicht möglich ist.

2) vgl. oben S. 44 - 47.

Erfahrungswerten des Verfassers mindestens 70 K-Bytes. Es kommt hinzu, daß es bei jedem Einplanungsschritt die Arbeit des Systems verzögert, da es erst aktiv werden kann, wenn vom Betriebssystem ein Job als erledigt gemeldet wurde oder eine bestimmte Zeitspanne verstrichen ist. Der nächste freigegebene Job kann erst nach dem Ende des Optimierungs-Programms den Routinen des Betriebssystems zur Ausführung übergeben werden.

Das zweite Verfahren nimmt die Belegungsplanung für die gesamte Planungsperiode im voraus vor. Bei ihm kehren sich die Vor- und Nachteile der ersten Lösung um. Bedingt durch Jobabbrüche und die Probleme der Bestimmung der Verweilzeiten, würde der Plan wahrscheinlich bereits nach einigen Stunden seine Aussagefähigkeit verlieren. Änderungen in der Verfügbarkeit der Kapazität, die z.B. durch die Anwendung des Time-Sharing oder der Datenfernverarbeitung hervorgerufen werden können, müssen im voraus zeitlich genau fixierbar sein und genauestens eingehalten werden, um die Gültigkeit des Plans nicht negativ zu beeinflussen. Da dieses Verfahren aber nur einmal durchgeführt werden muß, ermöglicht es eine Belegungplanung, die mit sehr geringen Kosten verbunden ist.

Bedenkt man, daß die Jobablaufplanung und die Belegungsplanung nur dann sinnvoll einzusetzen sind, wenn die Einsparungen, die durch sie ermöglicht werden, größer sind als die Kosten, die durch ihren Ablauf entstehen, ergibt sich ein Bewertungsmaßstab für die Belegungsplanung. Sie darf nicht so aufwendig sein wie der erste Vorschlag - so bestechend der Gedanke einer kontinuierlichen Belegungsplanung auch sein mag - , sie darf andererseits nicht so starr sein wie der zweite Vorschlag. Die besten Ergebnisse für eine optimale Belegungsplanung dürfte ein Verfahren versprechen, das einen Kompromiß zwischen beiden Extremen darstellt. Bei ihm wird die gesamte Planungsperiode in Abschnitte aufgeteilt, deren Länge von der Verwendbarkeit des alten Belegungsplans abhängig ist. Eine Neuplanung würde z.B. erfolgen, wenn Kapazitätsänderungen durch die Datenfernverarbeitung oder die Time-Sharing-Verarbeitung abzusehen sind, wenn Jobabbrüche eine weitere Verwendung des alten Belegungsplans nicht mehr gestatten, oder wenn die tatsächlichen Verweilzeiten

der Jobs sich zu weit von den Schätzwerten entfernen. Der Abruf der neuen Planung, die von den geänderten Größen ausgehen muß, wird vom Konsoloperator vorgenommen, dem auch die Aufgabe obliegt, dem Planungsprogramm eventuell aufgetretene Kapazitätsänderungen einzugeben. Jobprioritäten, Bereitstellungs- und Abgabetermine können bei dieser Planungsart ohne Schwierigkeiten berücksichtigt werden. Die Kompromißlösung erhält sich zum großen Teil die Flexibilität der Simultanplanung, umgeht aber die mit ihrer Durchführung verbundenen hohen Kosten. Deshalb soll dieser Vorschlag der Belegungsplanung in dieser Arbeit zugrundegelegt werden[1].

Ein optimaler Belegungsplan ist dadurch gekennzeichnet, daß die einzelnen Jobs unter Beachtung vorgegebener Restriktionen (Jobpriorität, Bereitstellungs- und Abgabetermine, Jobabhängigkeiten, gleiche Eingabedateien) so kombiniert werden, daß das Optimierungsziel, in diesem Fall die Minimierung der Gesamtverarbeitungszeit, erreicht wird. Die Verfahren, die die Lösung dieses Problems ermöglichen, lassen sich in exakte Verfahren, die mit Sicherheit zur Optimallösung führen, und in heuristische Verfahren einteilen, die die optimale Reihenfolge nicht mit Sicherheit finden, sondern meist nur Lösungen aufzeigen, die in der Nähe des Optimums liegen. Die Güte einer Lösung, die mit Hilfe eines heuristischen Verfahrens gefunden wird, läßt sich dabei meist nur durch Abschätzung bzw. Erfahrungswerte bestimmen.

In Abb. 24 sind die in der Literatur[2] am häufigsten genannten Methoden zur Lösung dieser Reihenfolgeprobleme im engeren Sinn[3] aufgezeigt. Die in den

---

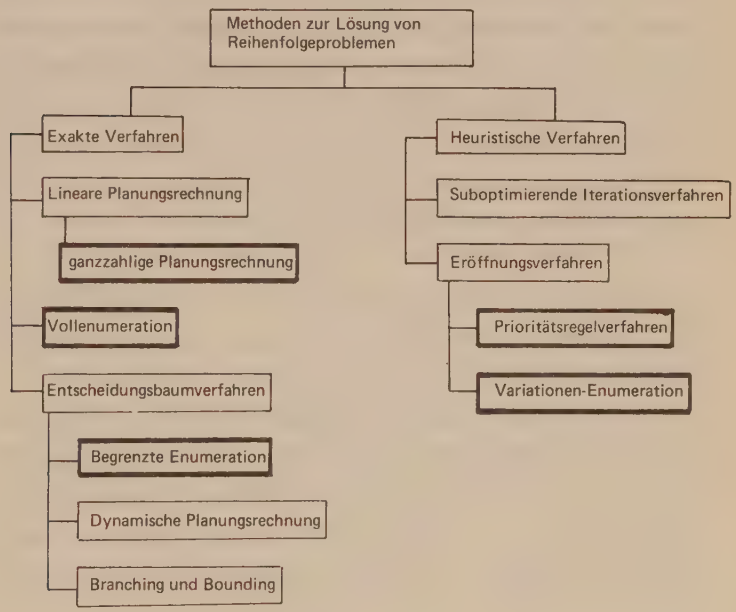
1) Ein diesem Vorschlag entsprechendes Programm zur Belegungsplanung wurde vom Verfasser an Testbeispielen bereits erprobt. Es wird Anfang 1973 zur Steuerung eines Rechenzentrums eingesetzt werden.

2) vgl. dazu u.a. H. MÜLLER-MERBACH, Optimale Reihenfolgen, Berlin 1970, und K. GEWALD, K. KASPER und H. SCHELLE, Netzplantechnik - Methoden zur Planung und Überwachung von Projekten, Band 2: Kapazitätsoptimierung, München 1972.

3) Wie auf S. 107 ausführlich dargelegt, gehört zu den Reihenfolgeproblemen im engeren Sinn beispielsweise nicht die Aufstellung des Jobablaufplans mit Hilfe eines der geschilderten Rangrechnungsverfahren.

stark umrandeten Kästchen aufgeführten Methoden werden in den beiden folgenden Abschnitten 5.2 und 5.3 ausführlich behandelt, wobei ihre Eignung für die Aufstellung optimaler Belegungspläne im einzelnen zu überprüfen ist. Die übrigen Verfahren, von deren Anwendung auf ein praktisches Problem der hier vorliegenden Art dem Verfasser nichts bekannt geworden ist, sollen nur kurz umrissen werden.

Abb. 24 : Methoden zur Lösung von Reihenfolgeproblemen



5.2 Exakte Verfahren

Die exakt arbeitenden Verfahren, bei denen mit Sicherheit eine Optimallösung für das Reihenfolgeproblem gefunden wird, lassen sich in drei Gruppen unterteilen. Der ersten Gruppe sind die Verfahren zuzuordnen, die sich zur Lösung des Problems mathematischer Methoden bedienen. Zur Anwendung kommt dabei ein Ansatz der Linearen Planungsrechnung . Da es sich bei der Belegungsplanung für Großrechenanlagen je-

doch um ein Problem handelt, für das nur ganzzahlige Lösungen gültig sind, muß in einem zweiten Schritt die Ganzzahligkeitsbedingung in das lineare Programm eingefügt werden. Die ganzzahlige Planungsrechnung liefert dann allerdings in der Regel nicht das "tatsächliche" Optimum, sondern nur ein in der Nähe des Optimums liegendes ganzzahliges "Suboptimum". In einigen Fällen gibt es überhaupt keine ganzzahlige Lösung des Problems[1]. Bereits an dieser Stelle ist zu diesen Verfahren zu sagen, daß die Aufstellung der Restriktionen und die Festlegung ihrer Zielfunktion nur mit erheblichem zeitlichen Aufwand möglich sind. Außerdem lassen sich nur relativ kleine Probleme in einer vertretbaren Rechenzeit lösen. MÜLLER-MERBACH stellt zu den Verfahren der ganzzahligen Planungsrechnung fest: "Probleme mit mehr als zwanzig ganzzahligen Variablen bereiten häufig schon einige Schwierigkeiten. Wenn die Zahl der Variablen mit Ganzzahligkeitsbedingung größer als vierzig bis fünfzig wird, übersteigt der Rechenaufwand fast immer das wirtschaftlich vertretbare Maß."[2]

In der zweiten und dritten Gruppe der Verfahren, von denen hier die Vollenumeration und die Begrenzte Enumeration ausführlicher erläutert werden sollen, werden alle Lösungsmöglichkeiten errechnet, die optimale Lösung wird ausgewählt und der Wert der Zielfunktion für diese Lösung bestimmt. Die Schwierigkeiten dieser Verfahren beruhen einerseits auf der sehr großen Zahl verschiedener Lösungen und liegen andererseits in dem zur Generierung aller zulässigen Lösungen notwendigen Verfahren. Auch diese Verfahren sind für Größenordnungen, wie sie in der Praxis auftreten, nicht anwendbar. MÜLLER-MERBACH[3] schätzt die Rechenzeit für die Lösung des Reihenfolgeproblems in einem vollständigen Graphen[4] mit 20 Knoten und mehr als $6{,}5 \cdot 10^{18}$ einzelnen Bauschritten

---

1) vgl. dazu K. GEWALD, K. KASPER, H. SCHELLE, a.a.O., S. 56, und H. MÜLLER-MERBACH, Operations Research, Berlin 1969, S. 366.

2) H. MÜLLER-MERBACH, Operations Research, a.a.O., S. 366 - 367.

3) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 32.

4) In einem vollständigen Graphen ist jeder Knoten direkt mit jedem anderen Knoten verbunden.

im Rahmen der Vollenumeration auf über 2 Millionen Jahre. Selbst wenn es gelingt, durch eines der Entscheidungsbaumverfahren die Rechenzeit auf 1 Milliardstel zu reduzieren, beträgt sie immer noch 18 Stunden.

Für die Belegungsplanung ist die Anzahl möglicher Lösungen immer nur für einen gegebenen Jobablaufplan und für gegebene Systemkapazitäten bestimmbar. Die Ermittlung des optimalen Belegungsplans mit Hilfe eines exakten Verfahrens ist bei kleineren Jobblöcken mit ca. 30 Jobs heute bereits in einer vertretbaren Zeit möglich. Für größere Jobblöcke sind die bisher bekannten Verfahren zur Zeit jedoch noch nicht geeignet. Um die Festlegung der Nebenbedingungen sowie der Zielfunktion zu verdeutlichen, sollen die Verfahren nur im Ansatz kurz erläutert werden[1].

Für die ganzzahlige, lineare Planungsrechnung wird der Ansatz von WIEST[2] skizziert, der als Zielfunktion die Minimierung der Gesamtverarbeitungszeit bei beschränkt verfügbaren Kapazitätstypen anstrebt[3]. Er bedient sich in seinem Ansatz sog. "0 - 1 Variabler", die den Zustand der Jobs in einem bestimmten Zeitpunkt beschreiben. Die Variable x_{jt} nimmt den Wert 1 an, wenn sich der Job j im Zeitpunkt t in der Ausführungsphase befindet, sie nimmt den Wert 0 an, wenn Job j nicht ausgeführt wird. Man erhält damit die erste Nebenbedingung $0 \leq x_{jt} \leq 1$, wobei x_{jt} ganzzahlig sein muß.

---

1) Auf die Durchrechnung eines konkreten Beispiels bei der Darstellung der exakten Verfahren soll in dieser Arbeit jedoch verzichtet werden, weil die Größenordnung des Belegungsplanungsproblems eine Lösung mit den zur Zeit bekannten Verfahren ohnehin nicht gestattet, und die Durchrechnung eines kleinen Beispiels für eine praktikable Lösung des Problems keine Erkenntnisse vermitteln kann.

2) J.D. WIEST, The scheduling of large projects with limited resources, PH.D. Dissertation, Carnegy Institute of Technology, Pittsburgh 1963.

3) Die Terminologie des WIEST'schen Ansatzes wird im folgenden nicht verwendet, sondern es wird die für die Belegungsplanung eingeführte Terminologie benutzt.

Summiert man für einen gegebenen Job j die Werte der Variablen x_{jt} vom Zeitpunkt 1 bis zum Ende des Belegungsplans T auf, erhält man die Verweilzeit VZ des Jobs j im System (Nebenbedingung 2):

$$\sum_{t=1}^{T} x_{jt} = VZ_j \quad (j = 1, \ldots, n)$$, wobei n die Anzahl der Jobs in der

Planungsperiode ist.

Bereits beim Aufbau des Jobablaufplans wurde festgelegt, daß ein Job während seiner Ausführung nicht unterbrochen werden darf[1]. Da Nebenbedingung 2 auch dann Gültigkeit besitzt, wenn ein Job beliebig oft unterbrochen wird, muß eine dritte Gruppe von Nebenbedingungen die ununterbrochene Erledigung der Jobs sicherstellen. Dies geschieht durch die Ungleichung

$$VZ_j \cdot x_{jt} - VZ_j \cdot x_{j(t+1)} + \sum_{t'=t+2}^{T} x_{jt} \leq VZ_j$$

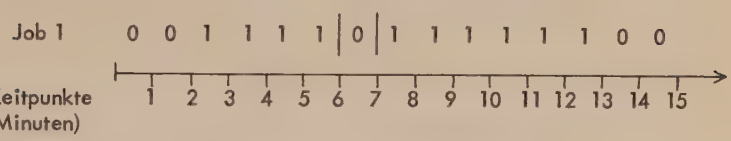
für $(j = 1, \ldots, n)$ und $(t = 1, \ldots, T)$.

An Hand des Beispiels in Abb. 25 soll dieser Zusammenhang deutlich gemacht werden.

---

1) vgl. dazu S. 95.

Abb. 25 : Beispiel einer unterbrochenen Jobausführung



0 : Unterbrechung des Jobs

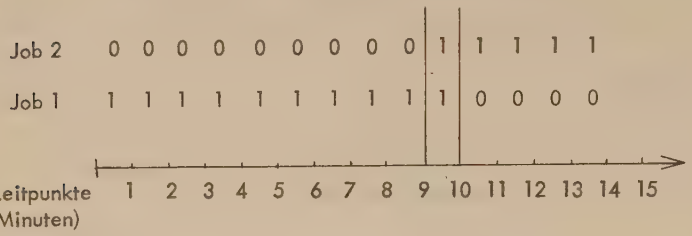
1 : Ausführung des Jobs

Wird Job 1 nach 6 Minuten für eine Minute unterbrochen, gilt die Ungleichung nicht mehr, denn die Differenz $VZ_j \cdot d_j - VZ_j \cdot d_{j(t+1)}$ wird in diesem Fall nicht gleich Null, so daß die linke Seite der Ungleichung größer wird als die Dauer des Jobs auf der rechten Seite: $10 \cdot 1 - 10 \cdot 0 + 6 > 10$.

Alle bisher festgelegten Nebenbedingungen beziehen sich auf den Job als einzuplanende Einheit. In einer vierten Gruppe von Nebenbedingungen müssen nun noch die Abhängigkeiten zwischen den Jobs festgelegt werden, denn ein Job j darf erst eingeplant werden, wenn seine sämtlichen Vorjobs i erledigt sind bzw. er keinen Vorjob besitzt. Die Formulierung dieser Restriktion fällt bei der Belegungsplanung nicht schwer, da als einzige Anordnungsbeziehung zwischen aufeinanderfolgenden Jobs die sog. Normalfolge in Frage kommt[1]. Die vierte Nebenbedingung lautet: $VZ_i \cdot x_{it} \leq \sum_{t'=1}^{t-1} x_{it'}$, , wobei $(t = 1,\ldots,T)$ und $(j = 1,\ldots,n)$ läuft. Würden zwei Jobs, die abhängig voneinander sind, überlappt eingeplant, würde die Restriktion verletzt, wie Abb. 26 zeigt.

1) Es handelt sich hier um einen in der DIN-Norm 69900 festgelegten Ausdruck für eine "Ende - Anfang" - Beziehung, die den Beginn eines Jobs zum Ende des Vorjobs in Abhängigkeit setzt. Diese Anordnungsbeziehung ist typisch für die Belegungsplanung.

Abb. 26 : Beispiel zur Darstellung von Jobabhängigkeiten in einem Modell der ganzzahligen, linearen Optimierung



Job 1 ist in diesem Beispiel Vorjob zu Job 2, seine Verweilzeit beträgt 10 Minuten. Job 2 dürfte also frühestens nach Erreichen des Zeitpunktes 10 beginnen. Da er bereits zum Zeitpunkt 10 eingeplant ist und $x_{2,10}$ den Wert 1 hat, ergibt die linke Seite der Ungleichung den Wert 10, die rechte Seite jedoch nur den Wert 9, und d.h., daß die Abhängigkeit nicht beachtet wurde.

Weitere Jobabhängigkeiten, die jedoch nicht von vorneherein genau festlegbar sind, treten nur bei der Belegungsplanung auf und sind bei WIEST nicht erwähnt. Es handelt sich um die Jobs k, die gleiche Eingabedateien anfordern wie ein betrachteter Job j und deshalb nicht parallel zueinander erledigt werden können, obwohl sie logisch unabhängig voneinander sind[1]. Diese Abhängigkeiten müssen in einer fünften Gruppe von Nebenbedingungen in der Form $x_{jt} + x_{kt} \leq 1$, für $j \neq k$ angegeben werden. Die Ungleichung stellt sicher, daß zwei Jobs j und k, die gleiche Eingabedateien benötigen, nicht parallel eingeplant werden.

Die sechste Gruppe von Nebenbedingungen stellt schließlich den Bedarf b an Kapazitätselementen s (z.B. Magnetband-, Magnetplattenspeichereinheiten und Kernspeicher-Regions) für einen gegebenen Zeitpunkt t der verfügbaren Kapazität a je Systemelement s zu diesem Zeitpunkt gegenüber, wobei die verfügbare Kapazität auf keinen Fall über-

1) vgl. dazu S. 103.

schritten werden darf. Diese Restriktionen lauten

$$\sum_{i=1}^{n} b_{si} \cdot x_{it} \leq a_{st} \quad , \text{ wobei } (t = 1,\ldots,T) \text{ und } (s = 1,\ldots m) \text{ ist.}$$

Die Berücksichtigung von Jobprioritäten, Bereitstellungs- und Abgabeterminen kompliziert das System der Nebenbedingungen zusätzlich in nicht unerheblicher Weise, ohne daß diese Restriktionen hier im einzelnen aufgeführt werden sollen.

Als Zielfunktion dieses ganzzahligen Optimierungsmodells legt WIEST eine geometrische Reihe fest, die jedem Zeitpunkt t+1 ein größeres Gewicht verleiht als einem Zeitpunkt t. Die Zielfunktion ist zu minimieren. Sie lautet:

$$\text{Min! } 1 \cdot \sum_{i=1}^{n} x_{it} + 4 \cdot \sum_{i=1}^{n} x_{i(t+1)} + 16 \cdot \sum_{i=1}^{n} x_{i(t+2)} + \ldots + R_T \sum_{i=1}^{n} x_{ir},$$

wobei $(t = 1,\ldots,T)$, $R_1 = 1$ und $R_t = 4 \cdot R_{t-1}$ ist. Durch die Gewichtung der einzelnen Reihenglieder erreicht WIEST, daß die einzelnen Jobs und damit auch das Ende des Belegungsplans möglichst wenig zum Projektende hin verschoben werden[1].

Zur Lösung eines solchen ganzzahligen, linearen Optimierungsmodells sind verschiedene Verfahren entwickelt worden. So bestimmt beispielsweise GOMORY[2] zunächst mit Hilfe eines Simplexverfahrens[3] die Lösung des linearen Programms ohne die Ganzzahligkeitsbedingung zu beachten. Erhält er ein nicht-ganzzahliges Ergebnis, werden iterativ neue Nebenbedingungen in das Modell eingefügt (sog. "cutting planes"), mit

---

1) An Stelle der von WIEST vorgeschlagenen Gewichtung könnte ebensogut eine andere Reihenentwicklung gesezt werden.

2) R.E. GOMORY, All-Integer Programming Algorithm, IBM Research Center, Research Report RC - 189, 1960
Drs., An Algorithm for Integer Solutions to Linear Programs, Recent Advances in Mathematical Programming, R. GRAVES and P. WOLFE, Editors, New York, 1963.

3) siehe dazu H. MÜLLER-MERBACH, Operations Research, a.a.O., S. 90 - 130.

denen das Modell wieder nach der Simplexmethode durchgerechnet wird. Durch das Einfügen neuer Restriktionen wird dabei der Lösungsbereich solange durch Abtrennung nicht-ganzzahliger Lösungen eingegrenzt, bis sich eine ganzzahlige Lösung ergibt, falls eine solche überhaupt existiert.

Neben dem Cutting-Plane-Verfahren können auch Entscheidungsbaumverfahren, und zwar Verfahren des Branching und Bounding, zur Lösung eines ganzzahligen Optimierungsmodells eingesetzt werden. Diese Verfahren gehen auf Arbeiten von LAND und DOIG[1] zurück, sind aber u.a. von DAKIN[2] wesentlich verbessert worden. Sie ermitteln mit Hilfe der Simplexmethode eine Lösung des Problems und legen dann jeweils für die nicht-ganzzahligen Variablen ganzzahlige Unter- und Obergrenzen fest, mit denen das Modell erneut durchgerechnet wird, bis eine ganzzahlige Lösung gefunden wird. Hat beispielsweise nur eine Variable x_1 bei der ersten Durchrechnung den Wert $x_1 = 1{,}6$ ergeben, wird die Rechnung mit $x_{1u} = 1$ und $x_{1o} = 2$ wiederholt, wobei die nicht zulässige Lösung jeweils ausgeschlossen wird.

Als Ergebnis der Durchrechnung erhält man für das Modell von WIEST Werte für die einzelnen x_{it}-Variablen. Durch ein einfaches Zusatzprogramm zu dem für die Optimierungsrechnung notwendigen Programm können aus diesen Angaben sowohl die Termine für den Belegungsplan als auch Belastungspläne für die einzelnen Systemkapazitäten gewonnen werden. Wie bereits erwähnt[3], kann die Belegungsplanung mit wirtschaftlichem Aufwand jedoch zur Zeit nicht mit einem ganzzahligen Optimierungsmodell gelöst werden. So benötigte WIEST beispielsweise für die Formulierung eines Problems mit 55 Tätigkeiten, die in vier Betriebsabteilungen durchgeführt werden, 5 275 Gleichungen mit 1 650 Variablen, ohne dabei die Gleichungen und Variablen zu berücksichtigen, die für die Ganzzahligkeit der Lösung zusätzlich eingeführt werden müssen.

---

1) A.H. LAND und A.G. DOIG, An Automatic Method of Solving Discrete Programming Problems, Econometrica 28, 1960, S. 497 - 520.

2) R.J. DAKIN, A Tree-Search Algorithm for Mixed Integer Programming Problems, The Computer Journal 8, 1965, S. 250 - 255.

3) vgl. dazu S.119, Fußnote 1.

Für "0 - 1" - Probleme, wie sie auch bei dem WIEST'schen Ansatz vorliegen, hat BALAS[1] einen besonderen Algorithmus entwickelt, der jedoch nicht näher erläutert werden soll, denn MÜLLER-MERBACH schreibt zur Effizienz dieses Vorgehens zu Recht[2]: "Diese Verfahren (die Entscheidungsbaumverfahren) kann man aber auch direkt auf die Reihenfolgeprobleme anwenden, ohne den Umweg über ein Modell der ganzzahligen Planungsrechnung einschlagen zu müssen. Die direkte Anwendung arbeitet meist wesentlich effizienter als die Anwendung auf das umformulierte und oft umständliche "0 - 1"-Problem".

Die Vollenumeration[3] stellt ein weiteres Verfahren zur exakten Berechnung der Optimallösung dar. Sie ist vom methodischen Aufbau her sehr einfach, da bei ihr alle möglichen Lösungen - unter Beachtung der angegebenen Restriktionen - aufgebaut und aus diesen die hinsichtlich des Optimierungsziels günstigste Lösung ausgewählt wird. Als Optimierungsziel könnte die Auswahl des Belegungsplans mit der geringsten Gesamtverarbeitungszeit festgelegt werden. Die besondere Schwierigkeit der Anwendung der vollständigen Enumeration bei der Belegungsplanung liegt neben der schon bei kleineren Problemen sehr schnell erreichten Grenze der Anwendbarkeit[4] auch darin, daß es bei Multiprogramming-Systemen erlaubt ist, mehrere Jobs parallel zueinander ablaufen zu lassen. Üblicherweise geht man bei der Vollenumeration davon aus, daß von einem Kapazitätstyp nur eine Einheit zur Verfügung steht, die also auch immer nur von einem Job belegt werden kann[5]. Entsprechend wird auch der Aufbau der Lösungs-

---

1) E. BALAS, An Additive Algorithm for Solving Linear Programs with Zero-One Variables, Operations Research, 13, No. 4, 1965, S. 517 - 546.

2) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 24.

3) vgl. dazu H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 25 - 27.

4) vgl. dazu S. 118.

5) In der Netzplantechnik bezeichnet man Tätigkeiten, die einen Kapazitätstyp verlangen, von dem nur eine Einheit zur Verfügung steht, als gebunden, die Tätigkeiten, die ihn nicht benötigen, als frei.

folgen bei der vollständigen Enumeration in der Regel sequentiell organisiert. Der wesentlich speicheraufwendigere parallele Aufbau der Folgen ist zwar prinzipiell ebenfalls möglich, setzt aber die Grenze für die Anwendbarkeit dieses Verfahrens noch weiter herab[1].

Die Entscheidungsbaumverfahren, die auch als verkürzte Enumerationsverfahren bezeichnet werden können, gehören zur dritten Gruppe der exakten Verfahren. Sie bauen wie die Vollenumeration alle möglichen Lösungen auf, brechen einen einmal beschrittenen Weg jedoch ab, wenn erkennbar wird, daß er zu keinem besseren als dem bisher günstigsten Ergebnis führt. Die Darstellung beschränkt sich hier auf das Verfahren der begrenzten Enumeration, da sie sich vor allem für Probleme "mit verhältnismäßig kurzen, aber breiten Entscheidungsbäumen, d.h. mit wenigen Stufen, aber vielen Zuständen pro Stufe"[2] eignet; genau diesen Aufbau besitzt in der Regel der Jobablaufplan.

Die Verfahren des Branching und Bounding sowie der Dynamischen Planungsrechnung reduzieren die Anzahl der Schritte zum Aufbau der Lösungsfolgen gegenüber der Vollenumeration bei Problemen dieser Art nur in sehr geringem Umfang, so daß sie für ihre Lösung kaum in Frage kommen.

Bei der begrenzten Enumeration berechnet man meist in einem ersten Schritt mit Hilfe eines der im nächsten Abschnitt 5.3 näher zu erläuternden Eröffnungsverfahrens eine günstige Ausgangslösung[3]. Das bietet den zweifachen Vorteil, daß man erstens den Enumerationsprozeß nicht mit einer zufällig gefundenen Lösung beginnt, die sehr ungünstig sein kann, und daß

---

1) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 26.

2) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 33.

3) H. MÜLLER-MERBACH, Ein Verfahren zur Planung des optimalen Betriebsmitteleinsatzes bei der Terminierung von Großprojekten, AWF-Mitteilungen, 1967, Heft 2, S. 5 - 10.

man zweitens bereits durch das Eröffungsverfahren die Optimallösung finden kann und den Enumerationsprozeß nicht mehr durchzuführen braucht. Die Prüfung, ob eine gefunde Lösung der Optimallösung entspricht, ist nicht immer einfach. Nur wenn die Kapazitäten in einem so großen Umfang zur Verfügung stehen, daß der auf Grund der Zeitrechnung aufgestellte Terminplan[1] sich hinsichtlich seiner Gesamtdauer mit dem durch die begrenzte Enumeration ermittelten Belegungsplan deckt, kann das Vorliegen der Optimallösung ohne Schwierigkeit erkannt werden. Die Dauer des Terminplans stellt die absolute Mindestgrenze für den Belegungsplan dar, weil er ausschließlich unter Beachtung der Jobabhängigkeiten und ohne Berücksichtigung der Kapazitätsrestriktionen aufgestellt wurde. Praktisch bedeutet dies, daß es möglich sein muß, die Jobs entweder ihren frühesten Terminen entsprechend einzuplanen oder ihre Termine nur im Rahmen der zur Verfügung stehenden Pufferzeit[2] zu verschieben. Bei der Belegungsplanung kommt dieser Fall aber praktisch nie vor, d.h. der Belegungsplan wird in der Regel eine höhere Dauer als der Terminplan aufweisen. Einen anderen Zeitraum für die Minimaldauer des Belegungsplans kann man nur festlegen, wenn man eine Vereinbarung darüber trifft, wieviele Jobs maximal parallel zueinander erledigt werden können. Arbeitet ein Rechenzentrum z.B. mit gleich großen, festen Region-Größen von 100 K-Byte und stehen für die Verarbeitung 400 K-Byte zur Verfügung, ist die Zahl der parallel ablaufenden Jobs auf 4 begrenzt. Der Belegungsplan mit der kürzestmöglichen Gesamtdauer entspricht dann dem Quotienten aus der Summe der Verarbeitungszeiten aller Jobs des Jobablaufplans dividiert durch die Anzahl der Regions, die für die Multiprogrammverarbeitung zur Verfügung stehen und in denen die Jobs parallel ablaufen können.

Die Frage, ob bei der begrenzten Enumeration eine aufgebaute Folge die Optimallösung oder zumindest ein besseres Ergebnis als das bisher günstigste erreichen kann, wird an Hand von zwei Kriterien überprüft. Das erste Kriterium vergleicht die bisher beste Lösung mit der Summe,

---

1) vgl. dazu S. 99.

2) vgl. dazu S. 97.

die aus dem Endzeitpunkt des soeben eingeplanten Jobs und der Mindestdauer gebildet wird, die bis zum Ende des Blocks, zu dem dieser Job gehört, noch vergehen muß. Diese Mindestdauern werden bereits bei der Zeitrechnung ermittelt, indem man den Zeitpunkt für das späteste Ende eines Jobs von der Gesamtdauer subtrahiert, die sich bei der Zeitrechnung für den entsprechenden Jobblock ergeben hat. Die Bestimmung der Mindestdauer verdeutlicht das folgende Beispiel, in dem die Dauern in Minuten angegeben sind:

Gesamtdauer eines Blocks bei der Zeitrechnung:		120
./. Spätester Endzeitpunkt des Jobs j in diesem Block:	SEZ_j =	40
Mindestdauer bis zum Blockende:		80

Es vergehen also mindestens noch 80 Minuten, bis der Block vollständig eingeplant ist.

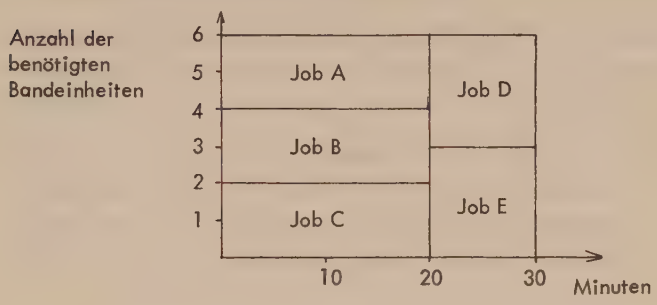
Das zweite Kriterium geht ebenfalls vom Endzeitpunkt des soeben eingeplanten Jobs aus und addiert auf diesen Wert die verbleibende Zeit, in der die einzelnen Kapazitätstypen mindestens noch durch die bisher nicht eingeplanten Jobs belegt sind. Der Vergleich dieser Summe mit der bisher besten Lösung ist relativ schnell durchzuführen, wenn von jedem Kapazitätstyp nur eine Einheit zur Verfügung steht, sie erweist sich jedoch als sehr zeitaufwendig und schwierig, wenn mehrere Einheiten verfügbar sind und jeder Job mehr als eine Einheit belegen kann, wie das folgende Beispiel zeigt:

5 Jobs A, B, .., E müssen noch erledigt werden. Es stehen 6 Bandeinheiten und 300 KB Kernspeicher zur Verfügung.

Die Jobs A, B und C im Rang 0 belegen für 20 Minuten je 2 Bandeinheiten und jeweils 100 KB, die Jobs D und E im Rang 1 belegen für 10 Minuten je 3 Bandeinheiten und jeweils 100 KB.

Wie Abb. 27 zeigt, können die 5 Jobs bereits nach 30 Minuten erledigt sein, wenn der Kernspeicher und/oder die Bandeinheiten auf Grund der Jobstruktur so optimal genutzt werden können wie es in diesem Beispiel möglich ist.

Abb. 27 : Beispiel zur Bestimmung der Restbelegungsdauer von Bandeinheiten für noch nicht eingeplante Jobs im Rahmen der begrenzten Enumeration



Die Bestimmung der Restbelegungsdauer ist bei Beachtung der Kapazitätsrestriktionen verschiedener Systemelemente und bei komplizierten noch einzuplanenden Jobstrukturen nach Meinung des Verfassers jedoch im allgemeinen nur durch eine grobe Abschätzung möglich, da die optimale Kombination der Jobs, wie sie hier angenommen wird, ja gerade das Ziel der Belegungsplanung ist. Die Restbelegungsdauer stellt bei Problemen mit Kapazitätstypen, von denen mehr als eine Einheit verfügbar ist, insofern kein Testkriterium für die Optimalität einer Lösung dar. Der Optimalitätstest kann hier nur an Hand des ersten Kriteriums erfolgen.

Ergibt sich im Verlauf des Enumerationsprozesses nach der Einplanung eines Jobs eine Gesamtbelegungsdauer, die ungünstiger ist als die bisher günstigste, wird der zuletzt eingeplante Job aus der aufgebauten Folge gestrichen und durch einen anderen ersetzt, für den wiederum der Optimalitätstest erfolgt, usw. Wird eine günstigere Lösung gefunden, fährt man mit dem Aufbau der Folge fort, d.h. man plant den nächsten Job ein, und setzt die Belegungsdauer der günstigeren Lösung als neue Überprüfungsgrenze ein. Die Lösung, die nach dem Aufbau aller Folgen den kleinsten Wert ergeben hat, stellt die Optimallösung dar.

Auch das Verfahren der begrenzten Enumeration kann laut MÜLLER-MERBACH [1] nur für kleine Probleme angewendet werden. Eine Aussage über die zur Lösung erforderliche Rechenzeit ist nicht möglich, da nicht von vornherein feststeht, wieviele Schritte zum Aufbau der optimalen Lösung notwendig sind.

Zusammenfassend kann man daher sagen, daß sich die exakten Verfahren für die Belegungsplanung bei den bisher zur Verfügung stehenden Lösungsmethoden und Rechnerkapazitäten n i c h t eignen, da sie sowohl in der Vorbereitungs- als auch in der Ausführungsphase zu aufwendig sind.

5.3 Heuristische Verfahren

Bei einer Anzahl von 150 - 200 einzuplanenden Jobs ist die Anwendung h e u r i s t i s c h e r V e r f a h r e n erforderlich. Diese Verfahren sind dadurch gekennzeichnet, daß sie zwar nicht unbedingt die optimale Lösung finden, aber in den meisten Fällen S u b o p t i m a aufzeigen, die in der Nähe des Optimums liegen. Ihr Hauptzweck ist darauf gerichtet, in einer Zeit von wenigen Minuten eine möglichst günstige Lösung anzugeben. Dabei ist es selbstverständlich, daß sie die aufgeführten Restriktionen in der gleichen Weise beachten wie dies auch bei den exakten Verfahren geschieht. Sie ersetzen die Zielfunktion durch Kriterien, die durch Überlegung gewonnen werden und das Lösungsverfahren wesentlich vereinfachen. In methodischer Hinsicht lassen sich bei den heuristischen Verfahren sogenannte Eröffnungsverfahren und suboptimierende Iterationsverfahren unterscheiden. Die E r ö f f n u n g s v e r f a h r e n [2] bestimmen auf direktem Weg in der Regel eine einzige Lösung, d.h. sie verglei-

---

1) vgl. dazu H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 200.

2) Der Name "Eröffnungsverfahren" kommt vermutlich daher, daß man diese Verfahren beispielsweise zur Bestimmung einer Hilfslösung für ein exakt arbeitendes Entscheidungsbaumverfahren anwendet, d.h. den Optimierungsprozeß "eröffnet".

chen nicht verschiedene Problemlösungen und suchen daraus die günstigste aus. Von einigen Autoren werden sie als die eigentlichen heuristischen Verfahren bezeichnet. Die suboptimierenden Iterationsverfahren versuchen schrittweise, eine bereits existierende Lösung zu verbessern, die in den meisten Fällen mit einem der Eröffnungsverfahren gewonnen wird. Die Optimallösung wird aber auch von ihnen häufig nicht erreicht.

Auf eine Reihe von Problemen, zu denen auch die Belegungsplanung zu zählen ist, sind bisher nur Eröffnungsverfahren angewendet worden. Der Verzicht auf suboptimierende Verfahren geschieht dabei aus den folgenden beiden Gründen[1]. Erstens sind viele der Probleme so umfangreich, daß die iterative Verbesserung einer Lösung zu rechenaufwendig ist, und zweitens ist die Wahrscheinlichkeit, daß man mit einem Eröffnungsverfahren eine von häufig mehreren, nahezu gleichwertigen Lösungen findet, so groß, daß man auf die Ermittlung einer nur geringfügig besseren Lösung verzichten kann.

Dieser Standpunkt mag zwar aus theoretischer Sicht unbefriedigend sein. Berücksichtigt man jedoch die verhältnismäßig kurze Zeitspanne, in der ein Belegungsplan Gültigkeit besitzt, so ist er für die Praxis aus wirtschaftlichen Gründen allein sinnvoll, denn die optimale Lösung der Belegungsplanung mit Hilfe eines exakten Verfahrens oder eines suboptimierenden Iterationsverfahrens ist nicht mehr sinnvoll, wenn sie mit überproportionalem Rechenaufwand und einem entsprechenden Zeitverlust erkauft werden muß.

Bevor einzelne Eröffnungsverfahren dargestellt werden, soll der Aktionsspielraum, der bei der Durchführung der Belegungsplanung besteht, noch kurz umrissen werden.

---

1) vgl. dazu H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 35.

In der Literatur[1] werden vier Möglichkeiten für eine Kapazitätsausgleichsplanung gesehen, nämlich 1.die zeitliche Verschiebung, 2.die zeitliche Streckung, 3.die zeitliche Stauchung und 4.die Unterbrechung von Tätigkeiten (Jobs).

Da die Ausführung der Jobs den Routinen des Betriebssystems unterliegt und vom Benutzer nicht gesteuert werden kann, kommt für die Belegungsplanung nur die Möglichkeit der zeitlichen Verschiebung in Frage. Wird dabei die Verschiebung eines Jobs um einen Zeitraum vorgenommen, der größer ist als seine gesamte Pufferzeit, ergibt sich stets ein Belegungsplan, der eine höhere Gesamtdauer ausweist als der auf Grund der ersten Zeitrechnung aufgestellte Terminplan.

Die Variationen-Enumeration, ein von FEHLER[2] vorgeschlagenes heuristisches Eröffnungsverfahren zur Kapazitätsausgleichsplanung, erlaubt in der bisher veröffentlichten Form die Einplanung nur eines Kapazitätstyps, von dem jeweils eine Einheit angefordert werden kann. Die Erweiterung auf mehrere Einsatzmitteltypen und eine gegebene Anzahl von Einheiten je Typ ist von FEHLER im Rahmen einer Dissertation geplant.

Bei der Variationen-Enumeration muß zunächst die Zeitrechnung durchgeführt werden, bei der für jeden Job neben den frühesten und spätesten Anfangs- bzw. Endterminen - ähnlich wie bei der begrenzten Enumeration - die Minimaldauer berechnet wird, die vom spätesten Endzeitpunkt eines Jobs bis zum Endzeitpunkt der Zeitrechnung des entsprechenden Jobblocks noch vergeht.

---

1) vgl. dazu u.a. J.P. VOIGT, Termin- und Kapazitätsplanung für Instandsetzungs- und Montageprojekte durch Netzplantechnik, Stahl und Eisen, 1971, Heft 20, S. 1125 - 1126

2) D. FEHLER, Die Variationen-Enumeration, Ein Näherungsverfahren zur Planung des optimalen Betriebsmitteleinsatzes bei der Terminierung von Projekten, Elektronische Datenverarbeitung, 1969, Heft 10, S. 479 - 483.

Die Variationen-Enumeration unterscheidet zwischen gebundenen Jobs, die das Kapaztitätselement benötigen, und den freien Jobs, die es nicht anfordern. Das Verfahren liefert für die gebundenen Jobs eine im Hinblick auf eine minimale Gesamtverarbeitungszeit möglichst günstige Reihenfolge, selbstverständlich werden aber auch die freien Jobs ihren Abhängigkeiten entsprechend berücksichtigt. Bei der Aufstellung des Belegungsplans werden zunächst alle einplanbaren freien Jobs fest terminiert. Ist daraufhin nur ein gebundener Job erreichbar, wird der Kapazitätstyp durch ihn belegt, ist mehr als ein gebundener Job einplanbar, wird eine Folge von zweien dieser Jobs gebildet und das frühestmögliche Ende des Belegungsplans bei einer angenommenen Einplanung dieser Folge berechnet. Das geschieht, indem man zum frühesten Endzeitpunkt jedes Jobs, der sich bei der dieser Folge entsprechenden Belegung des Kapazitätselements ergibt, die Minimaldauer addiert, die bis zum Ende des Belegungsplans mindestens noch vergeht. Der größere der beiden Werte gibt das frühestmögliche Ende bei Einhaltung dieser Folge an. Dann kehrt man die Jobfolge um, d.h. der Job, der zunächst an erster Stelle eingeplant war, rückt nun an die zweite Stelle und umgekehrt und berechnet für diese Umkehrfolge wieder wie oben das frühestmögliche Ende des Belegungsplans. Der Job beider Folgen, dessen Vorausstellung zu dem kleineren Planende geführt hat, wird entweder eingeplant, wenn kein weiterer gebundener Job erreichbar ist, oder es werden solange mit den anderen gebundenen Jobs Folgen und Umkehrfolgen gebildet, bis ein Job als endgültiger Sieger eingeplant werden kann[1]. Dann werden die Termine der eventuell neu erreichbaren freien Jobs festgelegt, bis mindestens wieder zwei gebundene Jobs zur Auswahl stehen, für die die Folgen und Umkehrfolgen gebildet werden, usw.

Die Aufstellung der Folgen und der entsprechenden Umkehrfolgen bietet den Vorteil, daß man von den einplanbaren gebundenen Jobs stets den Job

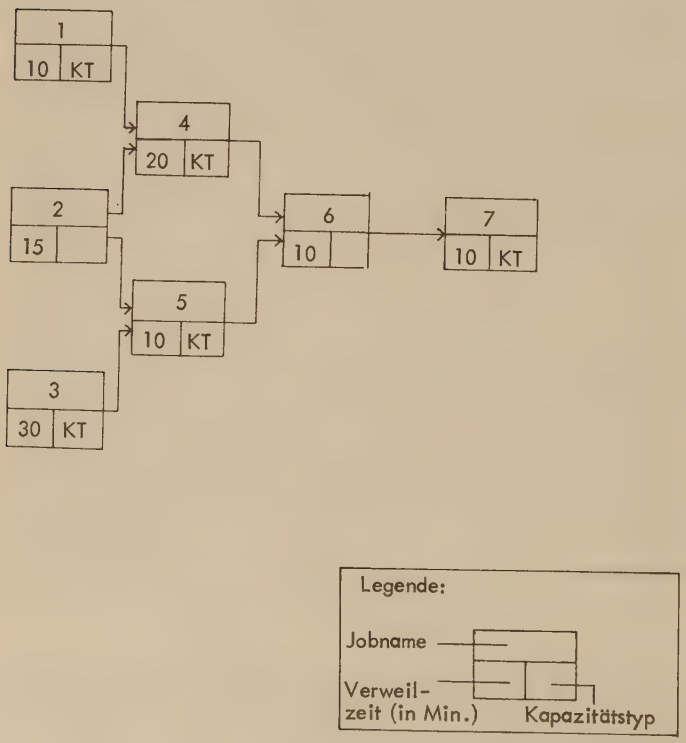
---

1) Die Bildung von Folgen und Umkehrfolgen für zwei Jobs bezeichnet man allgemein in der Kombinatorik als "Variationen der Klasse 2". Da alle möglichen Zweierfolgen gebildet werden, wird das Verfahren von FEHLER als "Variationen-Enumeration" bezeichnet.

auswählt, der die kleinste Gesamtverarbeitungszeit für den Belegungsplan erwarten läßt.

An Hand der in Abb. 28 gezeigten Jobstruktur und der Ergebnisse der Zeitrechnung aus Tab. 24 soll die Methode der Variationen-Enumeration noch einmal erläutert werden. Die mit "KT" gekennzeichneten Jobs fordern den zur Verfügung stehenden Kapazitätstyp an.

Abb. 28 : Jobstruktur für die Durchführung der Variationen-Enumeration



Tab. 24 : Zeitrechnung für die Jobstruktur aus Abb. 28

Jobname	Verweilzeit	Frühester Anfangs-Zeitpunkt FAZ	Frühester End-Zeitpunkt FEZ	Spätester Anfangs-Zeitpunkt SAZ	Spätester End-Zeitpunkt SEZ	Minimaldauer bis zum Blockende MD
1	10	0	10	15	25	35
2	15	0	15	10	25	35
3	30	0	30	0	30	30
4	15	15	30	25	40	20
5	10	30	40	30	40	20
6	10	40	50	40	50	10
7	10	50	60	50	60	0

Die Spalte MD enthält die Dauer, die vom spätesten Endzeitpunkt eines Jobs bis zum Projektende mindestens noch vergeht.

Zu Beginn der Belegungsplanung sind der freie Job 2, der vom Zeitpunkt 0 bis zum Zeitpunkt 15 eingeplant wird, und die gebundenen Jobs 1 und 3 erreichbar. Mit diesen Jobs wird die Folge 1-3 und die Umkehrfolge 3-1 gebildet, um herauszufinden, ob die Vorausstellung von Job 1 oder diejenige von Job 3 zu einem kürzeren Ende des Belegungsplans führt. Die Folge 1 - 3 ergibt für Job 1 eine Einplanung im Zeitraum 0-10 und für Job 3 eine Belegung von 10-40. Der frühestmögliche Endtermin des Belegungsplans bei dieser Folge entspricht dem Maximum von: $\max\{10+35, 40+30\} = \max\{45, 70\} = 70$, wobei 35 bzw. 30 die Minimaldauer bis zum Blockende angibt (vgl. Tab. 24). Bei der Folge 3 - 1 würde Job 3 von 0-30 und Job 1 von 30-40 eingeplant. Der frühestmögliche Endtermin wäre $\max.\{30+30, 40+35\} = 75$. Zunächst wird also Job 1 eingeplant, da er die kürzere Gesamtverarbeitungszeit von 70 Minuten erwarten läßt. Die nächste Folge ist 4 - 3 bzw. 3 - 4. Zu beachten ist, daß Job 4 auf Grund der Abhängigkeit zu Job 2 frühestens zum Zeitpunkt 15 eingeplant werden kann. Die Folge 4 - 3 ergibt ein frühestes Ende von 90, Folge 3 - 4 ein solches von 75. Job 3 wird eingeplant, usw. Tab. 25 zeigt die einzelnen Entscheidungssituationen auf, die zur Auswahl des entsprechenden Jobs geführt haben. Dabei bedeutet die Angabe beispielsweise der Jobfolge 3/4 und der zugehörigen frühestmöglichen Endtermine 75/90, daß die Folge 3 - 4 ein frühestes Ende des Belegungsplans nach 75 Minuten, und

die Umkehrfolge 4 - 3 ein solches nach 90 Minuten erwarten läßt. Auch wenn kein Parallel-Lauf zweier Jobs möglich ist, spielt die Einplanungsfolge also eine nicht unerhebliche Rolle. Die Auswahl von Job 4 bei Entscheidung 3 beruht auf dem kleineren frühesten Anfangstermin dieses Jobs gegenüber dem FAZ von Job 5. Die Jobs 5 und 7 werden auf Grund ihrer Abhängigkeiten eingeplant. Die optimale Reihenfolge, die man in diesem Beispiel für die Belegung des Kapazitätstyps erhält, ist also 1-3-4-5-7.

Tab. 25 : Entscheidungssituationen beim Verfahren der Variationen-Enumeration für das Beispiel aus Abb. 28

Entscheidung	Jobfolgen	Paare der Mindesttermine für das frühestmögl. Ende des Belegungsplans	Auswahl von Job	Kapazitätstyp belegt bis zum Zeitpunkt
1	1/3	70/75	1	10
2	3/4	75/90	3	35
3	4/5	80/80	4	50
4	5	80	5	60
5	7	80	7	80

Das Verfahren führt aber keineswegs immer zum Optimum. Nach Aussagen von MÜLLER-MERBACH [1] führt es dagegen fast immer zu besseren Ergebnissen als die Prioritätsregelverfahren. Numerische Erfahrungen mit grösseren Beispielen sind bisher allerdings noch nicht publiziert worden. Sicher zu sein scheint allerdings, daß die Variationen-Enumeration wesentlich aufwendiger ist als die Prioritätsregelverfahren. Dies gilt insbesondere auch für die Belegungsplanung, bei der man freie Jobs nicht kennt, da jeder Job mindestens einen Kapazitätstyp anfordert. Das bedeutet, daß für alle jeweils einplanbaren Jobs Folgen und Umkehrfolgen gebildet werden

1) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 199

müssen, und zwar unter zusätzlicher Beachtung der Kapazitätsrestriktionen. Eine abschließende Bewertung dieses Verfahrens sollte daher erst nach dem Bekanntwerden der Erweiterungen vorgenommen werden.

Die bekanntesten Eröffnungs-Verfahren zur Lösung von Reihenfolgeproblemen, die sich netzwerkmäßig aufgliedern lassen, basieren auf der Anwendung von Prioritätsregeln [1]. Die Prioritätsregelverfahren sortieren die Jobs vor oder während des Einplanungsprozesses nach vorgegebenen Kriterien, von denen erwartet wird, daß sie unter Beachtung der Kapazitätsrestriktionen das Optimierungsziel, vorzugsweise eine minimale Gesamtverarbeitungszeit, weitgehend erreichen. Ihr größter Vorteil liegt zweifellos in der Schnelligkeit begründet, mit der sie selbst bei umfangreichen Problemen eine gültige Lösung ermitteln, ohne daß dabei allerdings eine Aussage über die Güte der Lösung bzw. über die Abweichung dieser Lösung vom Optimum getroffen werden kann. In der Praxis[2] haben sich die Ergebnisse der Zeitrechnung - im Hinblick auf das Optimierungsziel - als "gute" Prioritätsregeln herausgestellt. Übertragen auf die Belegungsplanung bedeutet dies, daß man die Einplanung der Jobs nach ihren kleinsten oder größten frühestmöglichen Anfangs- oder Endtermin, ihrem spätestzulässigen Anfangs- oder Endtermin, oder nach ihrer kleinsten gesamten Pufferzeit vornimmt. Aber auch andere, in irgendeinem Zusammenhang zu den Jobs bzw. Kapazitätstypen stehende Größen können zu guten Einplanungsergebnissen führen. Z.B. möglich ist eine Vorrangregelung, die die Dauer der Jobs, die Höhe ihres Kapazitätsbedarfs

---

1) vgl. dazu u.a. K. GEWALD, K. KASPER, H. SCHELLE, Netzplantechnik, Band 2: Kapazitätsoptimierung, S. 80 - 114 und K. WEBER, Projektanalyse mit CPM, a.a.O., S. 239 - 248.

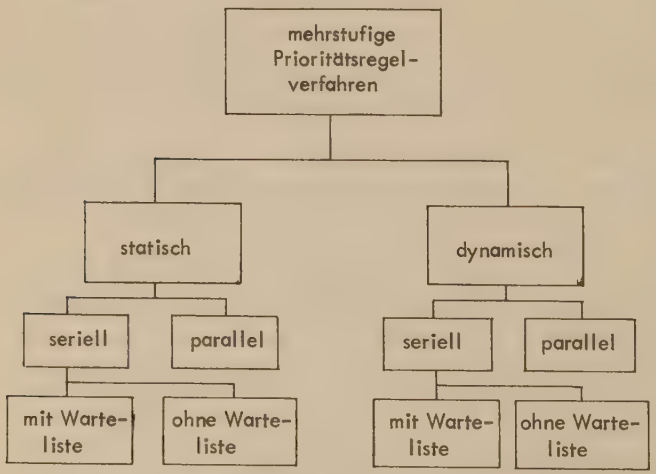
2) Die Kapazitätsausgleichsverfahren, die auf der Verwendung von Prioritätsregeln basieren, werden vor allem im Rahmen der Netzplantechnik bei Bauprojekten, Reparaturvorhaben sowie Forschungs- und Entwicklungsaufträgen in einfachen Versionen eingesetzt. Da diese Vorhaben meist langfristiger Natur sind, hat sich die Verwendung von auf den Werten der Zeitrechnung aufbauenden Prioritätsregeln angeboten. In Kapitel 5.4 wird noch zu zeigen sein, daß sich die dabei gewonnenen Erfahrungen nur sehr begrenzt auf die Belegungsplanung übertragen lassen.

an bestimmten Kapazitätstypen, die Dauer des jeweiligen Jobblocks, Jobprioritäten, die Anzahl der Folgejobs bei den einzelnen Jobs oder Zufallszahlen als Einplanungskriterien ansieht.

Diese Aufzählung muß eventuell bei Bedarf erweitert werden, da eine Prioritätsregel bei bestimmten Problemen durchweg zu sehr guten, bei anderen dagegen nur zu durchschnittlichen Ergebnissen führen kann, so daß sich die Auswahl einer bestimmten Regel immer an der Aufgabenstellung orientieren muß [1].

Einen ebenso großen Einfluß wie die Auswahl der Prioritätsregel(n) kann auch die Auswahl des Einplanungsverfahrens auf die Güte des Belegungsplans haben. Abb. 29 zeigt die Untergliederungsmöglichkeiten bei den sog. mehrstufigen Prioritätsregelverfahren auf.

Abb. 29 : Einteilung der mehrstufigen Prioritätsregelverfahren



1) In Abschnitt 5.4 werden die Prioritätsregeln aufgeführt, die bei der Belegungsplanung zu guten Ergebnissen geführt haben.

Man bezeichnet ein Einplanungsverfahren als mehrstufig, wenn ihm eine Kombination von mehreren Prioritätsregeln zugrundegelegt wird. Bei einem dreistufigen Verfahren würde z.B. eine Prioritätsregel als Primär-, eine zweite als Sekundär- und eine dritte als Tertiärregel festgelegt. Der Zweck dieses Vorgehens besteht darin, daß durch die Kombination verschiedener Prioritätsregeln eine eindeutige Rangfolge der Jobs für die Einplanung bestimmt werden kann. Dabei werden die Jobs zunächst nach dem Primärkriterium zur Einplanung ausgewählt. Ergibt sich dabei, daß mehrere Jobs einplanbar wären, erfolgt die weitere Auswahl nach der Sekundärregel, und bei gleichem Ordnungsbegriff mehrerer Jobs auch bei dieser Regel erfolgt schließlich die Einplanung nach dem Tertiärkriterium. Nahezu alle komfortablen Kapazitätsausgleichsprogramme arbeiten mit diesen mehrstufigen Verfahren. Der Einfluß der Tertiärregel auf den Plan ist dabei allerdings meist nur marginal[1].

Die Untergliederung auf der ersten Ebene in Abb. 29 in statische und dynamische Verfahren bezieht sich auf die Verfahren, die sich die Ergebnisse der Terminplanung zunutze machen. Bei der Einplanung der Jobs können entweder die einmal vor Beginn des Einplanungsprozesses ermittelten Werte der Zeitrechnung, die ohne Rücksicht auf Kapazitätsrestriktionen bestimmt wurden, oder die sich erst während des Einplanungsprozesses ergebenden Zeitwerte benutzt werden, die die Verschiebung der Jobs auf Grund der Kapazitätsgrenzen berücksichtigen. Verfahren der ersten Art nennt man statisch, die der zweiten Art dynamisch.

Auf der zweiten Ebene werden die Verfahren in seriell und parallel arbeitende Verfahren unterteilt. Seriell nennt man ein Vorgehen, bei dem zunächst alle Jobs den Prioritäts-

---

1) Neben den mehrstufigen Einplanungsverfahren gibt es - hauptsächlich bei manuell durchgeführtem Kapazitätsausgleich - auch einstufige Prioritätsregelverfahren, bei denen die Sortierung der Jobs auf Grund nur einer Prioritätsregel durchgeführt wird. Da bei der Belegungsplanung die Einplanung der Jobs immer unter Zuhilfenahme einer Prioritätsregelkombination erfolgen muß, soll auf die einstufigen Verfahren nicht eingegangen werden.

regeln entsprechend sortiert werden, bei dem aber immer nur der Job eingeplant wird, der auf Grund der Prioritätsregeln an der Reihe ist. Einige dieser seriellen Verfahren bauen sich zusätzlich Wartelisten auf, in die Jobs übernommen werden, die zwar auf Grund ihrer hohen Priorität, nicht aber auf Grund ihrer Abhängigkeiten oder wegen fehlender Kapazitäten eingeplant werden können. Parallel arbeitet ein Einplanungsverfahren, wenn nur diejenigen Jobs der Reihenfolge nach geordnet und eingeplant werden, die zu einem bestimmten Zeitpunkt des Einplanungsprozesses unter Berücksichtigung ihrer Anordnungsbeziehungen einplanbar sind.

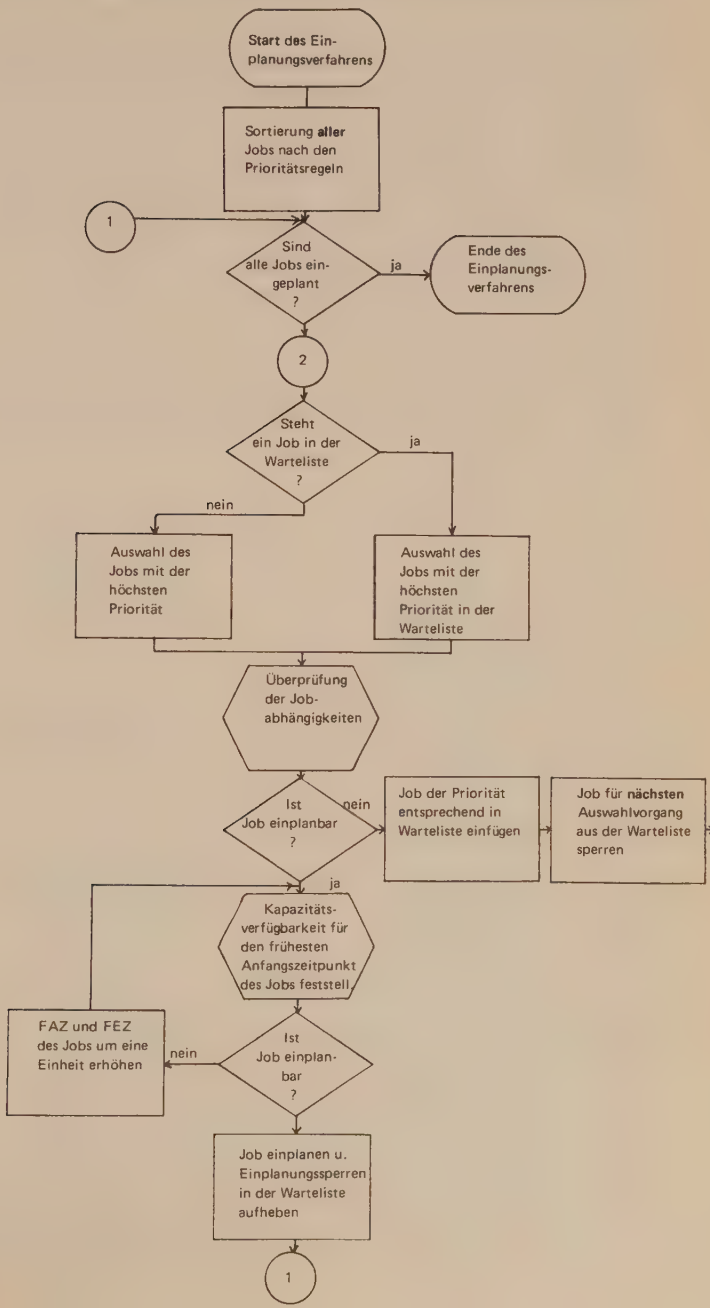
Ordnet man die Einplanungsverfahren nach dem Rechenaufwand und dem Grad der Kompliziertheit, mit der sie durchzuführen sind, so kann man sagen, daß die statischen Verfahren in der Regel einfacher arbeiten als die dynamischen, die seriell einplandenden einfacher sind als die parallel vorgehenden und die Verfahren ohne Warteliste einen einfacheren Aufbau besitzen als diejenigen mit Warteliste[1]. Das mehrstufige statisch-serielle Prioritätsregelverfahren mit Warteliste und das dynamisch parallele Verfahren sollen im folgenden ausführlich an Hand von Ablaufdiagrammen beschrieben werden. Die Unterschiede, die - verglichen mit diesen Verfahren - bei den anderen Verfahren bestehen, können dann ohne große Schwierigkeiten abgeleitet werden.

Abb. 30 zeigt das vereinfachte Ablaufdiagramm eines mehrstufigen statisch-seriellen Einplanungsverfahrens mit Warteliste. Es beginnt nach der Durchführung der Zeitrechnung, die nicht im Ablaufdiagramm enthalten ist, weil sie nicht zum eigentlichen Einplanungsprozeß gehört, mit der Sortierung aller Jobs nach den gewählten Prioritätsregeln. Durch diese Sortierung erhält jeder Job eine eindeutige Stellung innerhalb einer Tabelle, die es dem Verfahren gestattet, jeweils den Job mit der höchsten Priorität auszuwählen. Als erstes muß allerdings

---

1) vgl. dazu K. GEWALD, K. KASPER, H. SCHELLE, Netzplantechnik, Band 2: Kapazitätsoptimierung, a.a.O., S. 123 - 124.

Abb. 30 : Ablaufdiagramm eines mehrstufigen, statisch-seriellen Einplanungsverfahrens mit Warteliste

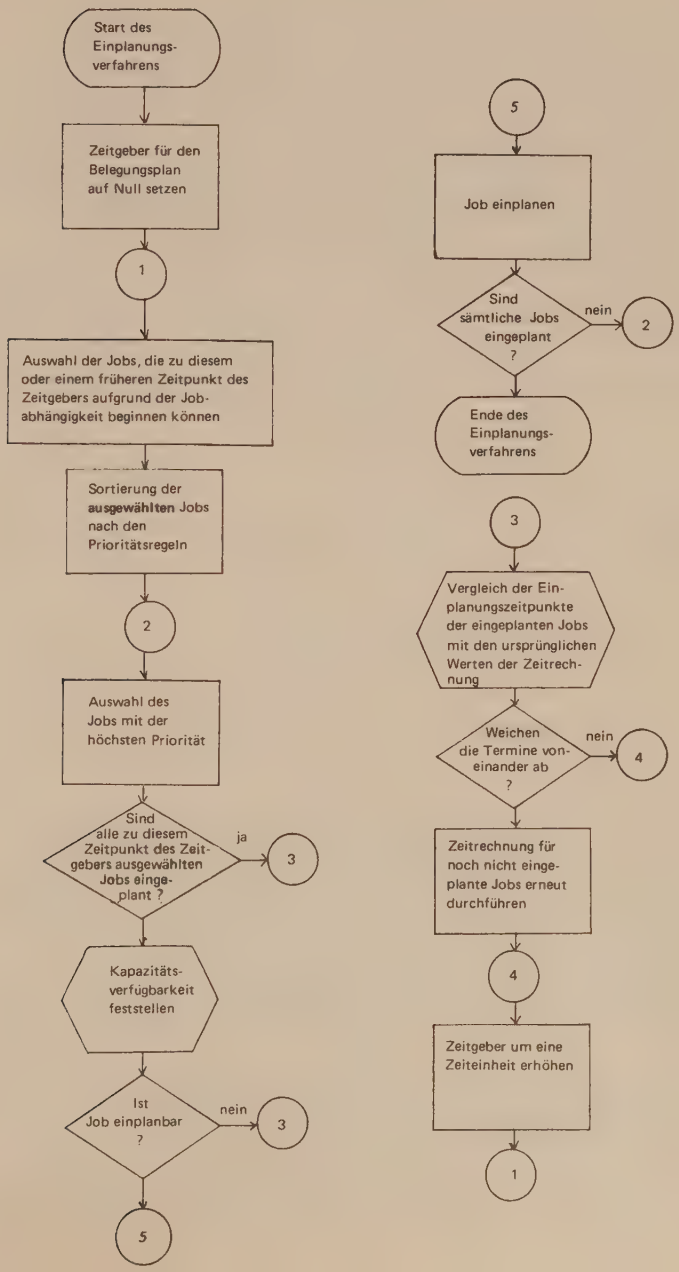


geprüft werden, ob der entsprechende Job auf Grund der Jobabhängigkeiten überhaupt einplanbar ist, d.h. ob seine Vorjobs erledigt sind, bzw. ob er keinen Vorjob besitzt. Hat die Überprüfung der Abhängigkeiten ergeben, daß ein Job noch nicht einplanbar ist, wird er in eine Warteliste übertragen, die ebenfalls nach den Prioritätskriterien geordnet ist. Ist ein Job dagegen einplanbar, versucht man, ihn zu seinem frühesten Anfangszeitpunkt einzuplanen. Wenn dies wegen der Kapazitätsbeschränkungen nicht möglich ist, verschiebt man ihn solange, bis genügend Kapazitäten verfügbar sind und plant ihn von diesem Zeitpunkt an ein. Bevor der nächste Job der nach Prioritäten geordneten Tabelle hinsichtlich seiner Einplanbarkeit untersucht wird, prüft man zunächst vorrangig die Jobs der Warteliste erneut auf ihre Einplanbarkeit, und wenn ihre Vorjobs inzwischen erledigt sind, plant man sie zuerst ein. Dieses Verfahren wird solange fortgesezt, bis alle Jobs eingeplant sind.

Das mehrstufige statisch-serielle Prioritätsregelverfahren ohne Warteliste arbeitet in der gleichen Weise, nur daß die Jobs, deren Vorjobs noch nicht eingeplant sind, hier nicht in eine Warteliste gesetzt werden, die ihre Abarbeitung zu einem späteren Zeitpunkt vorrangig macht. Sie werden vielmehr weiter in der "normalen", nach Prioritäten geordneten Tabelle geführt, was bedeutet, daß sich ihre Einplanung gegenüber dem Verfahren mit Warteliste stärker verzögern kann.

Bei dem in Abb. 31 aufgeführten Ablaufdiagramm für das mehrstufige dynamisch-parallele Einplanungsverfahren fällt auf, daß das Verfahren einen sog. Zeitgeber benutzt, der die einzelnen Zeitpunkte des Belegungsplans widerspiegelt und der im Verlauf der Einplanung solange um je eine Einheit erhöht wird, bis alle Jobs eingeplant sind. Der Zeitgeber ist notwendig, weil die parallel arbeitenden Verfahren nicht wie die seriellen Verfahren bereits zu Beginn des Einplanungsprozesses alle Jobs den Prioritätsregeln entsprechend sortieren, sondern nur die Jobs auswählen, die zu einem bestimmten Zeitpunkt des Belegungsplans (der durch den Zeitgeber repräsentiert wird) auf Grund ihrer Abhängigkeiten eingeplant werden können. Daraus folgt auch, daß die parallelen Ver-

Abb. 31 : Ablaufdiagramm des mehrstufigen, dynamisch-parallelen Einplanungsverfahrens



fahren keine Warteliste benötigen, in die ja gerade die wegen der Abhängigkeiten noch nicht einplanbaren Jobs übernommen wurden. Für den Startzeitpunkt werden demnach die Jobs ausgewählt, die keine Vorjobs besitzen, und in der Reihenfolge sortiert, die sich aus den Prioritätsregeln ergibt. Unter Beachtung der Kapazitätsbeschränkungen werden die ausgewählten Jobs dann in der Reihenfolge der Prioritätstabelle eingeplant. Jobs, die wegen der Kapazitätsbeschränkungen zu dem augenblicklichen Zeitpunkt des Zeitgebers noch nicht eingeplant werden konnten, und Jobs, die auf Grund ihrer Abhängigkeiten zusätzlich einplanbar sind, werden bei der Betrachtung des nächsten Zeitpunkts berücksichtigt. Das dynamische Moment kommt in den Einplanungsprozeß, wenn ein Job eingeplant wurde, dessen Einplanungstermine nicht mit den ursprünglichen Terminen der Zeitrechnung übereinstimmen. Es wird nämlich bei jedem Vorrücken des Zeitgebers, d.h. bei jedem neu betrachteten Zeitpunkt des Belegungsplans geprüft, ob eine solche Terminabweichung vorliegt, und wenn das der Fall ist, wird für alle noch nicht eingeplanten Jobs eine neue Zeitrechnung durchgeführt. Diese neuen Termine werden bei der Festlegung der Einplanungsreihenfolge für den nächsten Zeitpunkt zugrundegelegt. Die ständig neu ermittelten Werte der Zeitrechnung weichen vor allem bei Jobstrukturen, die den parallelen Ablauf vieler Jobs ermöglichen, in erheblichem Maß von den Werten der ersten Zeitrechnung ab und können erhebliche Änderungen in der Einplanungsreihenfolge der Jobs gegenüber einem statischen Verfahren bewirken.

Das mehrstufige statisch-parallele Einplanungsverfahren arbeitet von der Methode her in der gleichen Weise wie das dynamische, wählt also auch nur die Jobs zur Einplanung aus, die zu einem bestimmten Zeitpunkt eingeplant werden können, und sortiert diese Jobs den Prioritätsregeln entsprechend. Es entfällt bei diesem Verfahren aber eventuell bei jedem Zeitpunkt des Belegungsplans der neu betrachtet wird, die wiederholte Zeitrechnung, da bei der Auswahl der Jobs immer auf die Werte der ersten Zeitrechnung zurückgegriffen wird.

Das Vorgehen der beiden restlichen Verfahren, nämlich der mehrstufigen dynamisch-seriellen Verfahren mit bzw. ohne Warteliste, läßt sich aus den bisherigen Ausführungen nun leicht erläutern. Das mehrstufige dynamisch-serielle Einplanungsverfahren mit Warteliste ordnet zunächst alle Jobs den Prioritätsregeln entsprechend, wählt den Job mit der höchsten Priorität aus, überprüft, ob er auf Grund der Jobabhängigkeiten laufen kann, und plant ihn zum frühestmöglichen Zeitpunkt ein, wenn diese Bedingung erfüllt ist. Besitzt der Job noch unerledigte Vorjobs, wird er in die Warteliste übernommen, die auch bei diesem Verfahren stets vorrangig abgearbeitet wird. Die Warteliste wird gelöscht, wenn die Einplanung eines Jobs nicht zu seinem frühesten Anfangszeitpunkt erfolgen konnte. In diesem Fall muß für die noch nicht eingeplanten Jobs die Zeitrechnung wiederholt und eine neue Einordnung der Jobs nach den Prioritätsregeln vorgenommen werden (Dynamisierung des Einplanungsprozesses).

Das mehrstufige dynamisch-serielle Verfahren ohne Warteliste arbeitet ebenso wie das mehrstufige dynamisch-serielle Verfahren mit Warteliste, nur daß es die Jobs, die noch unerledigte Vorjobs besitzen, nicht in eine Warteliste übernimmt, sondern versucht, einen Job mit geringerer Priorität einzuplanen, dessen Vorjobs bereits erledigt sind.

Umfangreiche Untersuchungen über die Wirksamkeit der einzelnen Prioritätsregeln und Einplanungsverfahren liegen nach Wissen des Verfassers bisher noch nicht vor. Eine kleinere Untersuchung von MÜLLER-MERBACH [1] beschränkt sich auf Prioritätsregeln, die von den Werten der Zeitrechnung ausgehen[2]. Um zu einem guten Ergebnis, d.h. zu einer möglichst geringen Gesamtdauer eines Projektes zu kommen, wird allgemein nur vorgeschlagen, bei der Lösung von Reihenfolgeproblemen mit Prioritätsregelverfahren mehrere Berechnungen mit unterschiedlichen Kriterien durchzuführen und die beste Lösung auszuwählen.

---

1) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 201 ff; drs., Ein Verfahren zur Planung des optimalen Betriebsmitteleinsatzes bei der Terminierung von Großprojekten, a.a.O., S. 5 - 10.

2) Die Ergebnisse dieser Untersuchung werden in Abschnitt 5.4 erläutert.

5.4 Einige Erfahrungen mit Prioritätsregelverfahren

Die Ausführungen in Kapitel 5.2 haben deutlich gemacht, daß der Einsatz exakter Verfahren für die Belegungsplanung mit vertretbarem Aufwand zur Zeit nicht möglich ist und voraussichtlich in absehbarer Zeit auch nicht möglich sein wird. Das Hauptinteresse zukünftiger Untersuchungen wird sich neben der Weiterentwicklung dieser Verfahren in der nächsten Zeit vor allem auf die der jeweiligen Aufgabenstellung entsprechende Festlegung von Kriterien für die Auswahl geeigneter heuristischer Verfahren richten müssen.

MÜLLER-MERBACH hat verschiedene Prioritätsregeln in kleineren Testreihen untersucht und kommt dabei auch zu einer gewissen Wertung dieser Regeln[1]. Im folgenden sollen erstens die Ergebnisse dieser Testreihen kurz dargestellt werden, und zweitens sollen an einem Beispiel, das vom Verfasser mit dem PMS-IV-Programm der IBM durchgerechnet wurde, verschiedene Prioritätsregelkombinationen näher untersucht werden, die besonders für die Belegungsplanung einer EDV-Anlage gute Ergebnisse versprechen.

Eine Belegungs- oder Kapazitätsausgleichsplanung für mehrere Kapazitätstypen ist sinnvoll nur unter Einsatz einer elektronischen Datenverarbeitungsanlage möglich. Jede Hersteller-Firma bietet zur Zeit mindestens ein Verfahren zur Berechnung von Netzwerken und zur Durchführung der Kapazitätsausgleichsplanung an. Die Programme arbeiten durchweg mit einem der geschilderten Prioritätsregelverfahren, wobei bisher die Anwendung der statischen Verfahren überwiegt, da die dynamische Einplanung erst in jüngster Zeit aufgekommen ist. Es kann nicht Aufgabe dieser Arbeit sein, die Möglichkeiten der einzelnen Programmsysteme ausführlich darzulegen und sie einer kritischen Betrachtung zu unterziehen, da die meisten Systeme für die spezielle Problematik der Belegungsplanung für Großrechenanlagen ohnehin ungeeignet sind.

---

1) H. MÜLLER-MERBACH, Optimale Reihenfolgen, a.a.O., S. 201 - 204.

Die Untersuchungen von MÜLLER-MERBACH beschränken sich auf einstufige Verfahren, d.h. er gibt jeweils nur eine Prioritätsregel für die Bestimmung der Einplanungsreihenfolge vor. Über die Art des Verfahrens (serielle oder parallele Einplanung) macht er ebensowenig eine Aussage, wie über die sicher wichtige Frage nach der Problemstruktur (Anzahl der gebundenen Jobs in den einzelnen Rängen), so daß die Ergebnisse dieser Untersuchung nicht ohne weiteres verallgemeinert werden dürfen. Die Programme, die er verwendet hat, sind in FORTRAN-IV geschrieben. Die Netzwerke enthalten 30 - 400 Tätigkeiten, wobei jedoch maximal 30 Tätigkeiten an einen Kapazitätstyp gebunden sind. Folgende Prioritätsregeln sind von MÜLLER-MERBACH auf ihre Eignung untersucht worden:

1) kleinster frühester Anfangstermin (statisch),
2) kleinster spätester Anfangstermin (statisch),
3) kleinster frühester Endtermin (statisch),
4) kleinster spätester Endtermin (statisch),
5) kleinster frühester Anfangstermin für die erste Hälfte der gebundenen Vorgänge, kleinster spätester Endtermin für die zweite Hälfte (statisch),
6) kleinste Verweilzeit,
7) kleinste Pufferzeit (statisch),
8) kleinster frühester Anfangstermin (dynamisch),
9) kleinster frühester Endtermin (dynamisch) und
10) kleinster frühester Anfangstermin, wenn zu diesem Zeitpunkt der Kapazitätstyp verfügbar ist, geringste Verweilzeit, wenn der Kapazitätstyp noch belegt ist (dynamisch).

Die Rechenzeit, die benötigt wurde, um zu einer gültigen Reihenfolge zu kommen, lag auf einer Anlage des Typs IBM 7040 bei maximal 64 Sekunden, war jedoch meist wesentlich geringer (7 - 20 Sekunden).

Bei 21 untersuchten Netzwerken führte die Einplanung in neun Fällen zur Optimallösung, die jeweils mit Hilfe des Verfahrens der begrenzten Enumeration ermittelt wurde. Die Prioritätsregel 8 (dynamisch ermittelter kleinster frühester Anfangstermin) erbrachte die besten Lösungen, nämlich dreimal die Optimallösung, zehnmal die beste Näherungslösung und nur dreimal

eine Abweichung um mehr als 10 % von der Optimallösung. Die Prioritätsregeln 6 und 7 (kleinste Verweilzeit und kleinste Pufferzeit) führten zu den schlechtesten Ergebnissen, denn sie erreichten keinmal die optimale Lösung, keinmal die beste Näherungslösung und wichen 21- bzw. 19-mal um mehr als 10 % von der Optimallösung ab. Die statischen waren den dynamischen Einplanungsverfahren durchweg unterlegen, wobei höchstens die Prioritätsregel 1 eine Ausnahme darstellte, bei der viermal die Optimallösung, siebenmal die beste Näherung, aber auch siebenmal eine Abweichung um mehr als 10 % von der Optimallösung festgestellt wurde.

MÜLLER-MERBACH kommt zu dem Ergebnis, daß keines der Verfahren mit hinreichender Sicherheit die optimale Lösung ermittelt und daß man aus diesem Grunde jedes Problem mit mehreren Prioritätsregeln durchrechnen sollte. Da es bei der Belegungsplanung für eine EDV-Anlage neben einer möglichst geringen Gesamtverarbeitungszeit hauptsächlich auch auf eine schnelle Verfügbarkeit der Lösung ankommt, die Berechnung mehrerer Lösungen in der Regel also nicht möglich sein wird, bietet sich für die Belegungsplanung in Anlehnung an die Ergebnisse von MÜLLER-MERBACH die Verwendung eines dynamischen Prioritätsregelverfahrens an, bei denen trotz der geringen Rechenzeit die Abweichung von der Optimallösung nur selten größer als 10 % war.

Die Struktur der in der Praxis auftretenden Jobablaufpläne[1], bei denen sehr viele Jobs in den Rängen 0, 1 und 2 vorkommen, und die Notwendigkeit, Jobprioritäten sowie die Bereitstellungs- und Abgabetermine einzelner Jobs zu berücksichtigen, erlauben es nicht, die von MÜLLER-MERBACH untersuchten einstufigen Prioritätsregelverfahren für die Belegungsplanung einzusetzen. Bei diesen Verfahren würde nämlich die gleiche Größe des Prioritätskriteriums bei mehreren Jobs (z.B. gleicher frühester Anfangszeitpunkt) zur zufälligen Auswahl und Einplanung eines Jobs führen.

---

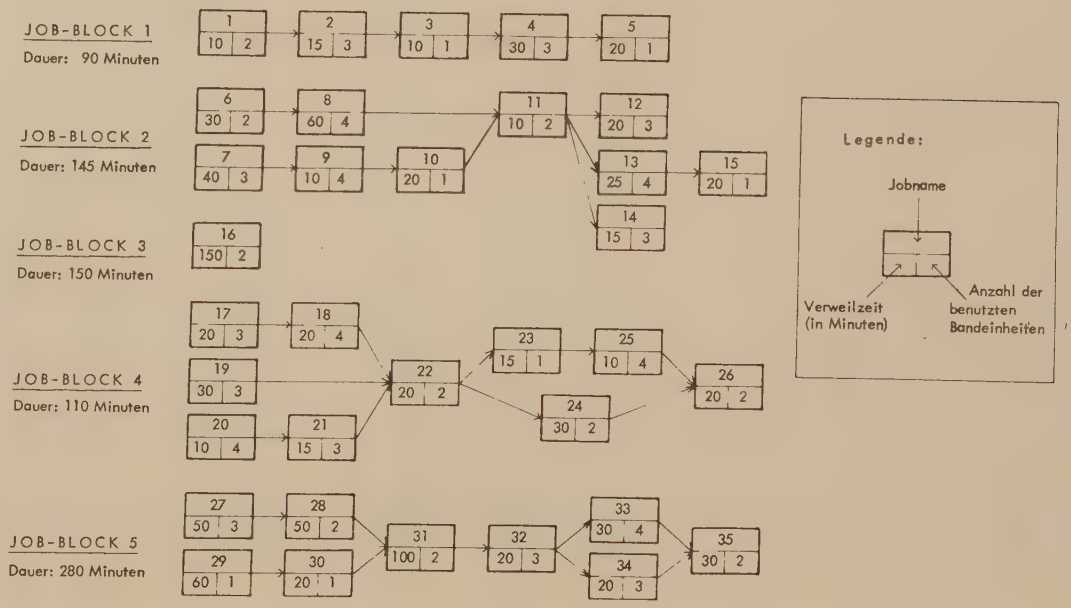
1) vgl. dazu oben S. 106.

Der Verfasser hat deshalb in einer eigenen Testreihe mit Jobstrukturen, die für Rechenzentren typisch sind, verschiedene Prioritätsregelkombinationen an Hand eines mehrstufigen Einplanungsverfahrens getestet. Die Tests wurden mit dem IBM-Programm zur Berechnung von Netzplänen PMS - IV durchgeführt, dessen Kapazitätsausgleichsteil RAP (Resource Allocation Processor) dem bekannten IBM-Kapazitätsausgleichsprogramm GRASP (General Resource Allocation and Scheduling Program) sehr ähnlich ist. Das Einplanungsverfahren von RAP arbeitet statisch-parallel. Es kann eine Primär-, eine Sekundär- und eine Tertiärregel angegeben werden, wobei wahlweise der höchste oder der niedrigste vorkommende Wert Priorität genießt.

Abb. 32 zeigt das mit 5 Jobblöcken und insgesamt 35 Jobs größte Test-Beispiel, an dem verschiedene Prioritätsregelkombinationen erprobt wurden. In diesem Beispiel sind aus Darstellungsgründen nur zwei Kapazitätstypen berücksichtigt worden: Magnetbandeinheiten und Kernspeicher. Sieben Bandeinheiten und ein 300 KB Kernspeicher standen für die Erledigung der Jobs zur Verfügung; jeder Job benötigte 100 KB, d.h. es durften maximal drei Jobs parallel ablaufen[1]. In Anhang 3 (s.S.222) ist das Ergebnis der Zeitrechnung für dieses Beispiel aufgeführt. Für die Zeitrechnung wurden die 5 Jobblöcke dabei aus programmtechnischen Gründen zu einem großen Jobblock zusammengefaßt, indem von einem fiktiven Startjob Scheinabhängigkeiten zu den Startjobs der einzelnen Jobblöcke definiert und entsprechende Scheinabhängigkeiten auch von den jeweils letzten Jobs der Jobblöcke zu einem gemeinsamen Endjob festgelegt wurden. Die kürzeste Belegungsdauer, die sich allein auf Grund der Jobabhängigkeiten ergibt, liegt bei 280 Minuten (Dauer des zeitlängsten Jobblocks 5), der kürzeste Belegungsplan, der unter Berücksichtigung der Kapazitätsrestriktionen durch die Anwendung verschiedener Prioritätsregelkombinationen erreicht wurde, hatte eine Gesamtdauer von 410 Minuten. Dieses starke Auseinanderfallen zwischen der theoretisch kürzesten Dauer (bedingt nur durch die Jobabhängigkeiten) und dem Ergebnis des Einplanungsprozesses ist für die Belegungs-

1) Auf die Angabe des benötigten Kernspeichers wurde in Abb. 32 verzichtet, da die Anforderung bei jedem Job gleich ist.

Abb. 32 : Test-Beispiel für die Durchführung der Belegungsplanung mit Hilfe verschiedener Prioritätsregelkombinationen



planung einer Großrechenanlage typisch. In der Regel wird der Belegungsplan eine mindestens vier- bis sechsmal so hohe Gesamtdauer ausweisen als es das Ergebnis der Zeitrechnung erwarten läßt. Für das Beispiel wurden die Prioritätsregeln 1. kleinster frühester Anfangszeitpunkt, 2. kleinster frühester Endzeitpunkt, 3. kleinster spätester Endzeitpunkt, 4. hohe Blockdauer, 5. kleinste gesamte Pufferzeit und 6. größte Verweilzeit zu jeweils dreistufigen Prioritätsregelkombinationen zusammengefaßt. Abb. 33 zeigt das Einplanungsergebnis für die Kombination 1. hohe Blockdauer, 2. niedrige gesamte Pufferzeit und 3. hohe Verweilzeit der Jobs[1]. Der Belegungsplan gibt die Belastung der Magnetbandperipherie wieder, läßt aber gleichzeitig auch erkennen, wieviele Jobs parallel zueinander erledigt werden sollen.

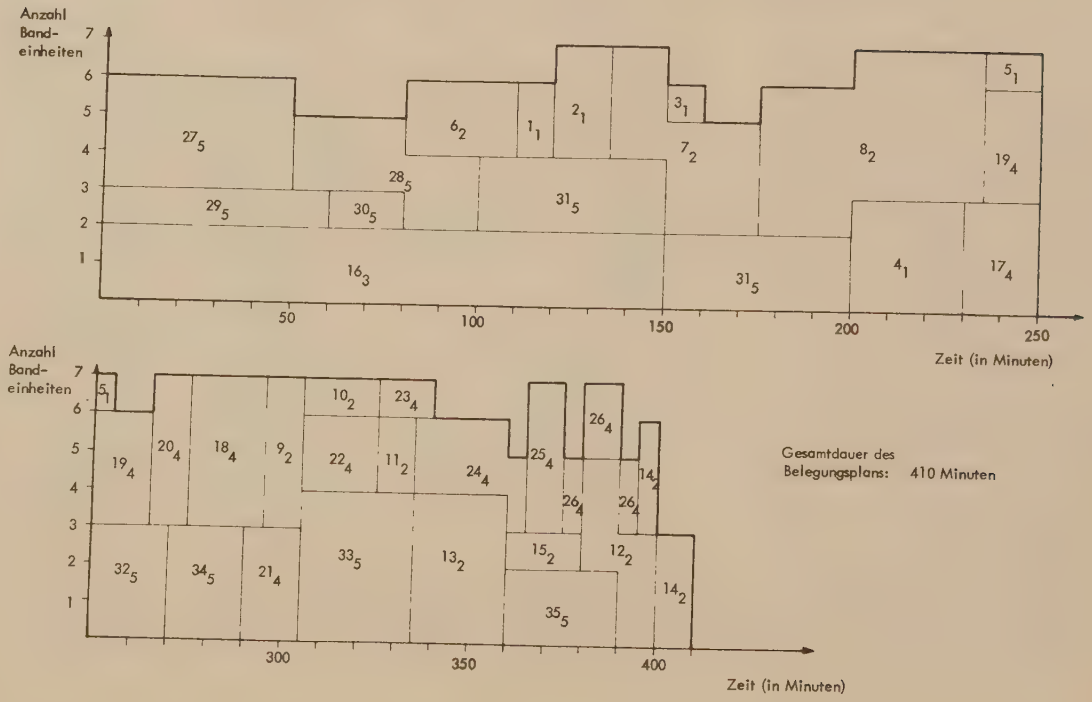
Bei den untersuchten Prioritätsregelkombinationen wurde in drei Fällen eine Gesamtdauer von 410 Minuten erreicht, zweimal trat ein Ergebnis von 420 Minuten und je einmal ein solches von 440 und 460 Minuten auf. Die genaue Einplanungsreihenfolge der Jobs bei ausgewählten Prioritätsregelkombinationen kann im Anhang den Abbildungen 4 - 9 entnommen werden. Sie sind entsprechend der Abb. 33 aufgebaut. Im Anhang ist auch im einzelnen vermerkt, zu welcher Gesamtdauer die jeweilige Kombination geführt hat.

Interessant ist nach Meinung des Verfassers, daß - und dieses Ergebnis trat auch bei den entsprechenden Testreihen der hier nicht näher erläuterten Beispiele auf - die schlechtesten Einplanungsergebnisse mit Prioritätsregeln erzielt wurden, die von den Ergebnissen der Zeitrechnung ausgehen. Dies gilt auch für die Primärregeln kleinster frühester Anfang und kleinstes frühestes Ende. In Übereinstimmung mit MÜLLER-MERBACH hat sich allerdings bei den Untersuchungen des Verfassers herausgestellt, daß das Kriterium "kleinster frühester Anfangszeitpunkt" innerhalb der Kriterien, die von den pro Job ermittelten Zeitpunkten ausgehen, die besten Belegungspläne erbrachte.

---

1) Die tiefer gesetzten Zahlen in Abb. 33 geben jeweils an, zu welchem Jobblock der eingeplante Job gehört.

Abb. 33 : Ergebnis der Einplanung für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination: 1. hohe Blockdauer, 2. niedrige gesamte Pufferzeit und 3. hohe Verweilzeit



Die besten Ergebnisse in allen Beispielen traten jedoch auf, wenn man die Regeln niedrige gesamte Pufferzeit, hohe Verweilzeit eines Jobs und hohe Blockdauer kombinierte. Hier zeigt sich eine deutliche Abweichung gegenüber den Ergebnissen von MÜLLER-MERBACH, die zweifellos auf die Struktur der Netzwerke zurückzuführen ist, die jeweils untersucht wurden. Aus der Tatsache, daß MÜLLER-MERBACH nur maximal 30 gebundene Tätigkeiten bei Netzwerken mit bis zu 400 Tätigkeiten zugelassen hat, läßt sich ableiten, daß die Zahl der parallel zueinander einplanbaren Tätigkeiten bei seinen Netzwerken nur gering war und daß in diesem Fall die auf der Zeitrechnung aufbauenden Prioritätsregeln die besseren Ergebnisse erbringen. Bei den vom Verfasser untersuchten Netzwerken konnten theoretisch sehr viele Jobs parallel zueinander ablaufen; auf Grund der kapazitiven Beschränkungen wurde jedoch häufig eine Verschiebung erforderlich, die es nicht erlaubte, einen Job zu seinen frühesten Zeitpunkten einzuplanen. In diesem Fall scheinen die angegebenen Prioritätsregeln die besseren Ergebnisse zu gewährleisten.

Sehr deutlich machte sich bei den untersuchten Jobstrukturen auch der Einfluß der Sekundärregel bemerkbar, während die Tertiärregel kaum Auswirkungen auf die Güte des Plans zeigte. Der Auswahl der Sekundärregel kommt demnach bei Problemstellungen, die mit der Belegungsplanung vergleichbar sind, eine entscheidende Bedeutung zu.

6 BESCHREIBUNG EINES HEURISTISCHEN VERFAHRENS FÜR DIE OPTIMIERENDE BELEGUNGSPLANUNG EINER IBM / 370 - 165 IN EINEM TECHNISCH / KAUFMÄNNISCH ORIENTIERTEN RECHENZENTRUM

6.1 Überlegungen zum organisatorischen Ablauf einer automatisierten Tages- und Belegungsplanung

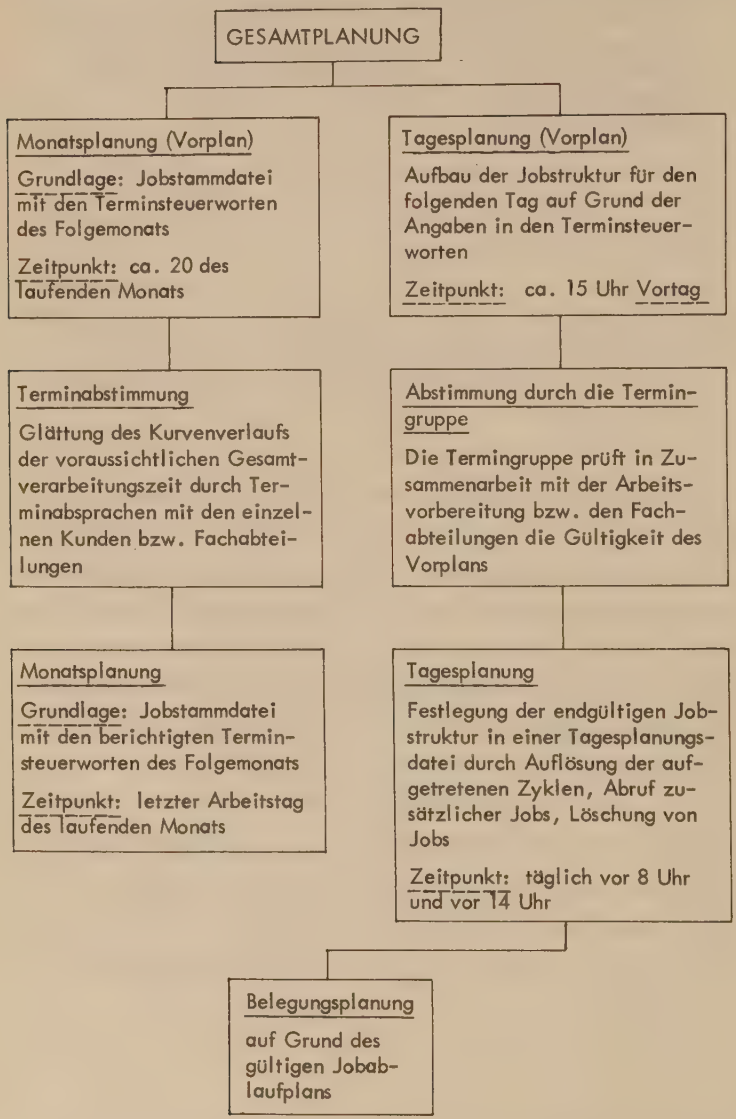
Mit der Einführung eines Planungssystems, als dessen letzte Ausbaustufe die automatisierte Belegungsplanung eines Großrechensystems anzusehen ist, verbinden sich gewisse Rationalisierungserwartungen. Ihre Realisierung hängt erstens davon ab, ob das gesamte Planungsverfahren [1] zu einer günstigeren Ausnutzung des Rechners führt als die von der Arbeitsvorbereitung durchgeführte manuelle Planung. Der Rationalisierungserfolg hängt zweitens davon ab, ob der organisatorische Ablauf innerhalb des Rechenzentrums so gestaltet werden kann, daß der Einsatz der Planungsverfahren überhaupt möglich wird. Besonders der Einfluß der Organisationsstruktur auf die erfolgreiche Einführung eines umfassenden Planungssystems kann nicht hoch genug eingestuft werden. Abb. 34 zeigt einen vom Verfasser erarbeiteten Vorschlag für das gesamte Planungssystem einer unter optimierenden Gesichtspunkten durchgeführten Stapelverarbeitung [2].

---

1) Damit sind hauptsächlich die Jobablaufplanung und die Belegungsplanung angesprochen.

2) vgl. oben S. 34.

Abb. 34 : Planungssystem für die Durchführung der Stapelverarbeitung



Die Monatsplanung [1] liefert der Leitung des Rechenzentrums durch eine Auswertung der Terminsteuerworte Informationen über die von einzelnen Firmen bzw. Fachabteilungen pro Tag angeforderte Rechenzeit. Zeigt der monatliche Vorplan starke Schwankungen bei den für jeden Tag kumulierten Gesamtverarbeitungszeiten, so werden Absprachen mit einzelnen Kunden zu Terminverschiebungen einzelner Jobs oder ganzer Arbeitsgebiete führen können, die einen geglätteten Kurvenverlauf der voraussichtlichen Gesamtverarbeitungszeit bewirken. Durch eine Berichtigung der Terminsteuerworte bei den entsprechenden Jobs in der Jobstammdatei bis zum letzten Arbeitstag des laufenden Monats steht dann zu Beginn des neuen Monats für die tägliche Planung eine aussagefähige Planungsunterlage zur Verfügung. Gelingt der Ausgleich nur unvollkommen, kann der Monatsplan zumindest eine wertvolle Entscheidungsunterlage für die Anmietung zusätzlicher Rechenzeit in einem Service-Rechenzentrum sein, da er die Belastungsspitzen bereits frühzeitig erkennen läßt.

Mit dem Ausdruck Tagesplanung ist hier der tägliche Aufbau des Jobablaufplans[2] gemeint, der lediglich die Abhängigkeiten zwischen den Jobs sowie ihre kapazitiven Anforderungen aufzeigt. Ein vorläufiger Tagesplan, der nur die Jobs berücksichtigt, die durch das Terminsteuerwort vorterminiert sind, sollte bereits am Vortag jeweils bis etwa 15 Uhr vorliegen, denn zu diesem Zeitpunkt weiß die Arbeitsvorbereitung des Rechenzentrums bereits, ob die vorterminierten Jobs tatsächlich am Planungstag laufen müssen. Die Gültigkeit des Vorplans wird von der Arbeitsvorbereitung in Zusammenarbeit mit den Fachabteilungen also schon am Vortag überprüft und gegebenenfalls korrigiert, indem aufgetretene Zyklen[3] aufgelöst, zusätzliche Jobs abgerufen und evtl. vorterminierte Jobs gelöscht werden. Im Laufe der Nacht wird auf Grund der Korrekturen der eigentliche Tagesplan erstellt, der am Planungstag bis 8 Uhr vorliegen muß. Dieser Plan behält bis mindestens 14 Uhr Gültigkeit und ist damit Grundlage für die

---

1) vgl. oben S. 64.

2) vgl. oben Kapitel 4.

3) vgl. oben S. 89.

Belegungsplanung , die eine möglichst optimale Zuordnung der einzelnen Systemelemente vornimmt und die Reihenfolge festlegt, in der die Jobs des Tagesplans erledigt werden. Gegen 14 Uhr wird bei Bedarf der Tagesplan neu aufgebaut, um den Fachabteilungen Gelegenheit zu geben, zusätzliche Jobs abzurufen. Die bis zu diesem Zeitpunkt erledigten Jobs werden in diesen zweiten Tagesplan natürlich nicht mehr aufgenommen, der dann bis zum nächsten Morgen gültig ist.

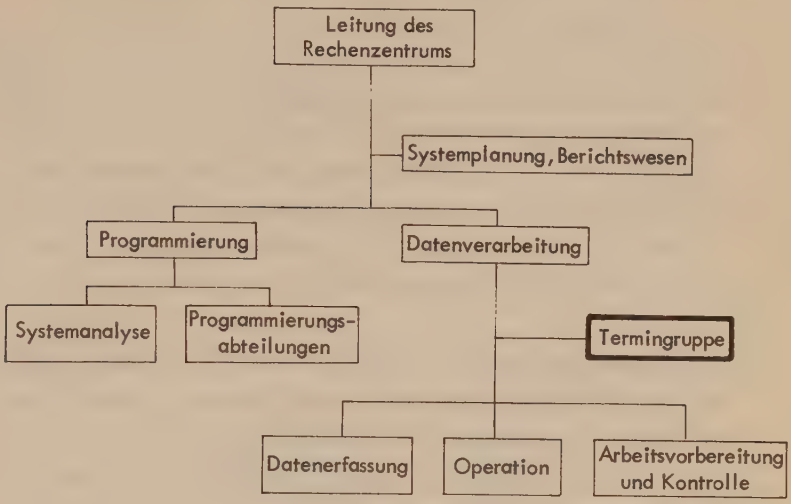
Die Belegungsplanung wird immer an Hand des neuesten Jobablaufplans vorgenommen. Über die Häufigkeit, mit der dieser Plan zu erstellen ist, lassen sich zur Zeit noch keine allgemeingültigen Aussagen machen. Erste Erfahrungen, die der Verfasser mit einem eigenen Programm zur Belegungsplanung gemacht hat, deuten darauf hin, daß der Belegungsplan ungefähr nach jeweils drei Stunden neu aufgebaut werden muß[1)].

Eines der Hauptprobleme in jedem Rechenzentrum ist mit Sicherheit in der Terminfestlegung der einzelnen Jobs zu sehen. Um dieses Problem im Sinn einer möglichst gleichmäßigen Maschinenbelastung und einer aussagefähigen Tagesplanung lösen zu können, empfiehlt es sich unbedingt, für die Jobterminierung eine eigene Termingruppe einzusetzen, die in Zusammenarbeit mit den Kunden des Rechenzentrums und auf Grund der in der Vergangenheit festgestellten Ist-Termine die Termingestaltung für die einzelnen Jobs vornimmt. Ihre mögliche Einordnung in die Organisationsstruktur eines Rechenzentrums zeigt Abb. 35.

---

1) Dieses in FORTRAN - IV geschriebene Programm basiert auf dem in Kapitel 6.3 erläuterten Einplanungsverfahren. Es wird zur Zeit vom Verfasser getestet und soll 1973 zur Steuerung eines Großrechenzentrums eingesetzt werden.

Abb. 35 : Organisationsstruktur eines Rechenzentrums



Der Aufgabenbereich dieser Termingruppe sollte neben der Termingestaltung auch die Jobannahme und die Erstellung des Tagesplans umfassen. Häufig ist leider festzustellen, daß diese Funktionen, die weitgehend voneinander abhängen, von verschiedenen Stellen nebenbei miterledigt werden. Daraus resultiert in der Praxis meist die ständige Änderung des Tagesplans, die es unmöglich macht, eine Belegungsplanung auch nur für einen kleinen Zeitraum durchzuführen. Unterstellt man die in Abb. 34 vorgeschlagenen Termine für die Tagesplanung, so muß der erste Tagesplan unter allen Umständen bis 14 Uhr Gültigkeit besitzen. Unter Berücksichtigung der erledigten Jobs und zusätzlicher Jobs kann zu diesem Zeitpunkt erneut die Feststellung der Jobstruktur vorgenommen werden, deren Gültigkeit dann bis zum Beginn der nächsten Planungsperiode gewährleistet sein muß, um eine erfolgreiche Belegungsplanung zu ermöglichen.

6.2 Erläuterung der Anlagenkonfiguration und der bei der Belegungsplanung berücksichtigten Systemelemente

Die vom Verfasser konzipierte Belegungsplanung wird im zentralen Rechenzentrum eines Konzerns der Stahlindustrie eingeführt[1]. Die Leistungen des Rechenzentrums entfallen zu etwa 53 % auf den kaufmännischen Bereich und zu etwa 47 % auf den technischen Bereich. Gearbeitet wird in der Regel im 3-Schicht-Betrieb, d.h. es stehen für alle anfallenden Aufgaben maximal 1 440 Minuten pro Region[2] und Tag zur Verfügung.

Zum besseren Verständnis der folgenden Abschnitte soll die Konfiguration des zur Verfügung stehenden Datenverarbeitungssystems kurz erläutert werden, ohne daß allerdings auf die technischen Daten der einzelnen Systemelemente näher eingegangen werden kann. In Abb. 36 ist die gesamte Konfiguration dargestellt[3]. Das Rechenzentrum verfügt über eine Datenverarbeitungsanlage des Typs IBM/370-165 und eine solche des Typs IBM/370-155. Auf der linken Seite von Abb. 36 ist die /370-165, auf der rechten Seite die /370-155 - getrennt durch die strichpunktierte Linie - mit den im Normalfall zugeordneten Einheiten dargestellt. Beide Anlagen werden in einem Systemverbund betrieben, da die IBM/370-155 hauptsächlich über Datenfernverarbeitung zur Optimierung des Fertigungsablaufs in verschiedenen Betrieben eingesetzt wird, und es deshalb sichergestellt sein mußte, daß bei einem hardware-bedingten Ausfall die Jobs zur Fertigungssteuerung auf einem zweiten System nahtlos weitergeführt werden können. Auf Grund dieses technischen Problems ergab sich die Notwendigkeit, die meisten Peripherieeinheiten wahlweise der einen oder der anderen Anlage zugänglich zu machen, um beispielsweise der /370-165 den Zugriff zu Da-

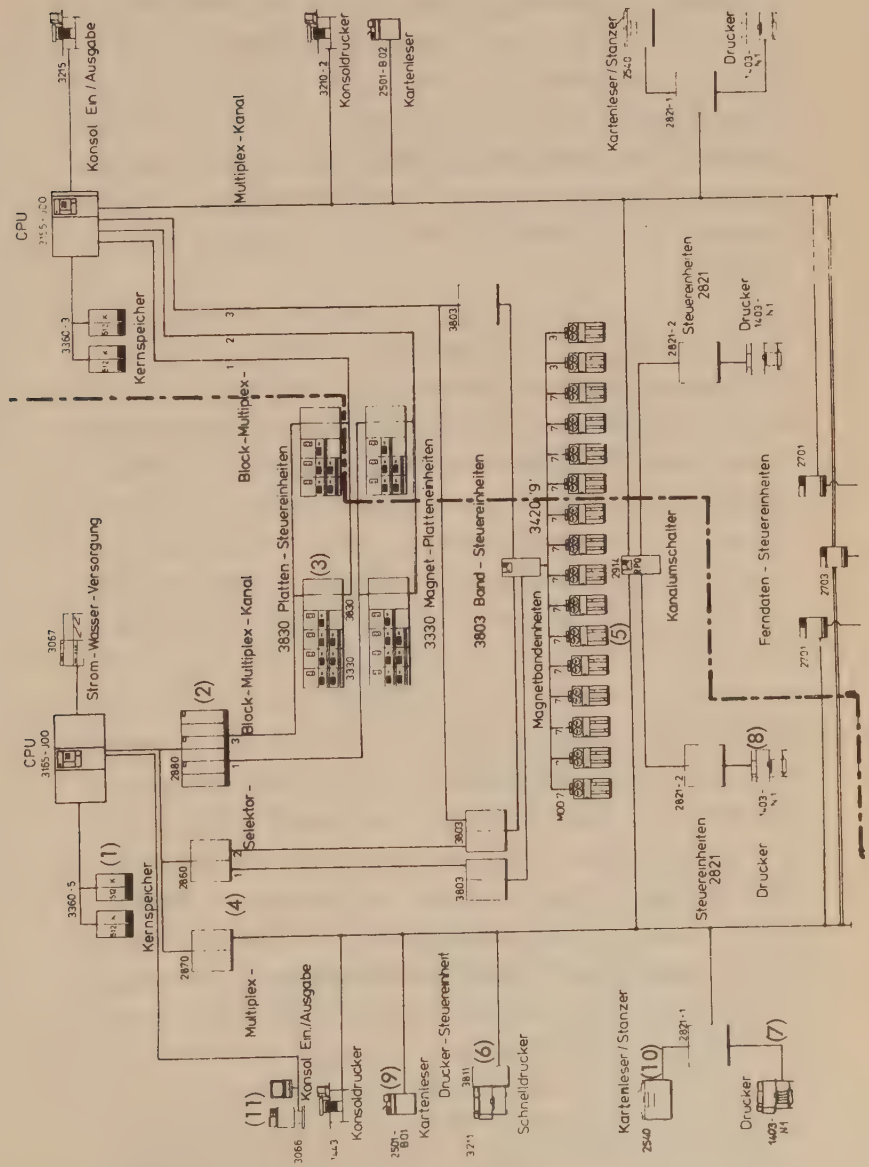
---

1) vgl. dazu oben S. 158, Fußnote 1.
Einige Testergebnisse sowie die Möglichkeiten dieses Belegungsplanungsprogramms werden in Kapitel 6.4 erläutert.

2) vgl. oben S. 19.

3) Die in diesem Abschnitt in Klammern angegebenen Zahlen im Text beziehen sich auf die Zahlen in Abb. 36.

Abb. 36 : Konfiguration des Systemverbundes bestehend aus einer Anlage des Typs IBM/370-165 und einer Anlage des Typs /370-155



teien zu ermöglichen, die die Fertigungssteuerungsprogramme benötigen. Die Belegungsplanung wird jedoch nur für die IBM/370-165 vorgenommen, die nahezu ausschließlich für die Stapelverarbeitung eingesetzt wird. Nur die für diese Verarbeitungsform notwendigen Systemelemente werden deshalb im folgenden erläutert. Im Regelfall sind der Anlage ganz bestimmte Peripheriegeräte fest zugeordnet, es kann jedoch geschehen, daß sie von der zweiten Anlage zusätzliche Einheiten erhält bzw. Einheiten abgeben muß. Bei der Belegungsplanung muß es deshalb möglich sein, die Kapazitätsverfügbarkeit der einzelnen Systemelemente im Zeitablauf variabel vorzugeben.

Das Modell 165 des IBM-Systems /370 zeichnet sich durch eine sehr hohe interne Verarbeitungsgeschwindigkeit aus. Die Zykluszeit des Pufferspeichers im zweistufigen Speichersystem beträgt ca. 80 Nanosekunden, die des Hauptspeichers liegt bei ca. 2 Mikrosekunden. Der Pufferspeicher übernimmt ständig Instruktionen und Daten aus dem Hauptspeicher, um der Zentraleinheit den Zugriff zum relativ langsamen Hauptspeicher zu ersparen. Untersuchungen haben ergeben, daß die Zentraleinheit in ca. 95 % aller Fälle die gewünschte Information im Pufferspeicher findet[1].

Die Hauptspeichergröße (1) des hier betrachteten Systems beträgt 1 024 Kilo-Bytes. Das Betriebssystem (OS-MVT) einschließlich verschiedener System-Tasks (Reader, Writer usw.) benötigt davon etwa 330 Kilo-Bytes. Dieser Bereich ist ständig belegt, d.h. für die Routinearbeiten stehen insgesamt ca. 694 Kilo-Bytes zur Verfügung. Eine Besonderheit des bei dieser Anlage installierten Betriebssystems besteht darin, daß während der Systemgenerierung die Größe der einzelnen Regions auf 104 Kilo-Bytes oder ein Vielfaches dieser Größe festgelegt ist. Diese Fixierung wählte man, weil sich bei umfangreichen Tests herausgestellt hat, daß die Kernspeicherausnutzung bei variabler Regiongröße eher schlechter als bei fester Regiongröße war. Hin-

---

1) IBM DEUTSCHLAND, Das IBM System /370 Modell 165, Form-Nr. 15822, S. 1

zu kam, daß sich die SORT und MERGE Dienstprogramme[1] der IBM bei dieser Größe als sehr schnell erwiesen und daß die Programmierung bei der zunehmenden Komplexität der Anwendungsprogramme einen deutlichen Trend zu größeren Kernspeicheranforderungen aufwies.

Die Festlegung der Regions auf 104 K-Bytes legt den ersten Parameter der Belegungsplanung fest. Der frei verfügbare Kernspeicher hat eine Größe von ca. 694 K-Bytes, d.h. maximal dürfen sechs Jobs parallel zueinander eingeplant werden.

Der Kernspeicher ist über zwei sog. Block-Multiplex-Kanäle (2), von denen jeder in der Grundausführung eine maximale Datenübertragungsrate von 1,5 Mio. Bytes pro Sekunde besitzt, mit zur Zeit drei Magnet-Platteneinheiten des Typs IBM 3330 verbunden. Eine dieser Magnet-Platteneinheiten kann wahlweise auch der IBM/370-155 zur Verfügung gestellt werden. Zwei Einheiten besitzen je 4 Laufwerke, eine Einheit verfügt über 8 Laufwerke. Jedes Laufwerk enthält einen Plattenstapel mit 12 Platten. Die Speicherkapazität eines solchen Plattenstapels beträgt 100 Millionen Bytes, d.h. die gesamte Speicherkapazität der Magnet-Platteneinheiten (3) beläuft sich auf 1,6 Milliarden Bytes. Diese 1,6 Milliarden Bytes stehen allerdings nicht voll als Arbeitsspeicherplatz für die Routinejobs zur Verfügung, denn die nicht kernspeicherresidenten Routinen des Betriebssystems, Programmbibliotheken des Anwenders, der Katalog für die Dateiverwaltung und auf den Platteneinheiten residente Dateien (vor allem Lohn-, Gehalts- und Auftragsbestandsdateien mit Stammdaten, die von sehr vielen Jobs benutzt werden) belegen ständig ca. 900 Millionen Bytes, so daß die für die einzelnen Jobs im Rahmen der Belegungsplanung verfügbare Kapazität 700 Mio. Bytes beträgt. Da die Speicherplatz-Anforderungen der einzelnen Jobs in sog. "Cylindern" de-

---

1) Das SORT-Programm sortiert numerische oder alphanumerische Begriffe von Dateien in auf- oder absteigender Folge, das MERGE-Programm mischt verschiedene Dateien nach einem beliebigen Ordnungsbegriff zu einer Datei.

finiert sind, muß auch die verfügbare Kapazität in dieser Maßeinheit ausgedrückt werden. Ein Plattenstapel ist in 404 Cylinder unterteilt, und das ergibt theoretisch einen verfügbaren Speicherplatz von 2828 Cylindern pro Zeiteinheit. Durch diese Größe ist die zweite Kapazitätsgrenze für die Belegungsplanung festgelegt. Die Kombination der Jobs darf nur so erfolgen, daß diese Grenze nicht überschritten wird. Die Bestimmung des für Arbeitsbereiche frei verfügbaren Speicherplatzes bereitet in der Praxis oft erhebliche Schwierigkeiten. Man sollte die Grenze auf keinen Fall zu knapp bemessen, da im Laufe einer Einplanungsperiode zusätzliche Dateien resident gemacht werden können, benutzte Arbeitsspeicherbereiche durch fehlerhafte Dispositionsangaben nach Beendigung eines Jobs weiter belegt bleiben, obwohl dafür keine Notwendigkeit gegeben ist, und auch die Programmbibliotheken ständig erweitert werden.

Besteht in einem Rechenzentrum eine ständige Engpaßsituation im Hinblick auf die Plattenspeicherzuordnung, so ist eine laufende Kommunikation zwischen allen Stellen erforderlich, die Einfluß auf die Belegung dieser Einheiten haben.

Die Verbindung zwischen den Magnetbandeinheiten und der Zentraleinheit bzw. dem Kernspeicher wird durch zwei Selektorkanäle (4) gewährleistet, deren Datenübertragungsrate bei jeweils maximal 1,3 Mio. Bytes/Sekunde liegt. Für die /370-165 sind im Normalfall zehn Magnetbandeinheiten (5) vorgesehen, die den dritten Kapazitätstyp darstellen, der bei der Belegungsplanung berücksichtigt wird. Sechs Einheiten stehen dem Modell 155 zur Verfügung. Die Bandeinheiten sind allerdings in ihrer Gesamtheit umschaltbar, d.h. sie können auch alle der /370-165 oder der /370-155 zugeordnet werden. Da die Bedienung der Bandeinheiten über eine eigene Konsolschreibmaschine erfolgt, die ausdruckt, welche Bandrollen auf welcher Einheit zu montieren oder zu demontieren sind, und da außerdem die Bereitstellung der Bandrollen vor der Erledigung eines Jobs stets gewährleistet ist, werden die Rüstzeiten für diese Einheiten bei der Belegungsplanung vernachlässigt.

Zur Standardausrüstung der /370-165 gehören ferner drei Drucker (6 - 8), zu denen bei erhöhtem Bedarf noch ein Drucker der /370-155 hinzugeschaltet werden kann, wobei auch die Druckerbelastung in den einzelnen Druckausgabeklassen direkt über eine eigene Konsolschreibmaschine ausgegeben wird. Die Berücksichtigung der Druckzeiten wird in Kapitel 6.3 ausführlich diskutiert, so daß sich ihre Behandlung an dieser Stelle erübrigt.

Die Eingabe der Lochkarten kann entweder über einen kleinen Kartenleser (9) vorgenommen werden, der vor allem zum Einlesen der Jobsteuerkarten dient oder über einen wesentlich leistungsfähigeren Leser (10), der vor allem für Datenkarten vorgesehen ist. Die Berücksichtigung der Ladezeiten ist nicht unproblematisch. Bei dem zur Zeit noch recht hohen Lochkartenaufkommen in dem betrachteten Rechenzentrum kann eine Außerachtlassung dieser Zeiten den Belegungsplan in Frage stellen. Die Ladezeiten können allerdings über die Angabe eines Bereitstellungstermins in einem Job umgangen werden. In naher Zukunft wird die Lochkartenerfassung aber voraussichtlich in den meisten Rechenzentren durch Datenerfassungssysteme abgelöst, die eine direkte Speicherung des Datenmaterials auf Plattenspeichereinheiten oder auf Magnetbändern ermöglichen. Es scheint daher sinnvoll zu sein, das Problem der Berücksichtigung von Ladezeiten bei der Belegungsplanung auszuklammern.

Die eigentliche Steuerung der Anlage erfolgt mit Hilfe einer alleinstehenden Konsole (11), die über einen Bildschirm verfügt, auf dem sich der Konsoloperator verschiedene Warteschlangen des Job Management anzeigen lassen und über eine Tastatur auch verändern kann[1].

---

1) vgl. oben S. 27.

6.3 Beschreibung des Belegungsplanungsverfahrens

Die Ausführungen in Kapitel 5 haben deutlich gemacht, daß die Belegungsplanung aus rechentechnischen Gründen nur mit Hilfe eines heuristischen Verfahrens durchgeführt werden kann. Der damit eventuell verbundene Verzicht auf die tatsächlich optimale Lösung wird fraglos durch eine grössere Schnelligkeit und Praktikabilität des Lösungsverfahrens aufgewogen. Die Bedeutung des Schnelligkeitsgesichtspunkts gilt um so mehr, wenn man bedenkt, daß der Belegungsplan mehrmals täglich erstellt werden muß[1]. Das Erfordernis einer praktikablen Lösung wird augenscheinlich, da in einem mehrschichtig betriebenen Rechenzentrum selbst ein Konsoloperator auf Grund der Tagesplanungsunterlagen die Möglichkeit besitzen muß, sich bei Bedarf einen neuen Plan ausgeben zu lassen. Dazu wird er aber nur in der Lage sein, wenn das Planungsverfahren einfach zu handhaben ist.

Das vom Verfasser konzipierte und auch bereits programmierte Verfahren ist den Eröffnungsverfahren zuzurechnen, und zwar handelt es sich um ein mehrstufiges, parallel einplanendes Prioritätsregelverfahren[2]. Die Notwendigkeit zur Entwicklung eines eigenen Verfahrens ergab sich, weil selbst die komfortabelsten Programmsysteme zur Kapazitätsausgleichsplanung in der Netzplantechnik maximal die Angabe von drei hintereinander zu schaltenden Prioritätsregeln gestatten, ein dreistufiges Prioritätsregelverfahren die Vielzahl der Bedingungen, die bei der Belegungsplanung zu berücksichtigen sind, aber nicht in ausreichender Weise beachten kann. Es kam hinzu, daß die dem Verfasser bekannten Eröffnungsverfahren[3] in der Handhabung der Daten zu umständlich waren und eine für die Belegungsplanung zu hohe Rechenzeit aufwiesen.

---

1) vgl. oben S. 158.

2) vgl. oben S. 138.

3) Beispielsweise das Programmsystem PMS (Project Management System)

Ziel der Planung ist es, die Zielgröße, nämlich die Gesamtverarbeitungszeit auf dem Anlagentyp IBM/370 Modell 165 zu minimieren, und zwar im Hinblick auf die in Abschnitt 6.2 festgelegten Systemelemente[1], d.h. es wird eine möglichst gute Belegung des Kernspeichers, der Magnetbandeinheiten sowie der Magnetplattenspeicherbereiche angestrebt. Die Zentraleinheit und die Ein-/Ausgabekanäle bleiben aus den bereits mehrfach genannten Gründen bei der Belegungsplanung ebenso unberücksichtigt wie die Bestimmung der Reihenfolge, in der auf den zur Verfügung stehenden Druckern die Ausgabelisten ausgegeben werden, denn bereits in Kapitel 2 [2] wurde deutlich, daß die Abarbeitung der Ausgabeklassen vom eigentlichen Jobablauf unabhängig erfolgt. Die Ausgabe kann nämlich sofort nach dem Ende eines Jobs erfolgen, sie kann aber auch mehrere Stunden hinausgezögert werden. In den meisten Fällen wird man die Ausgabe nach dem Prinzip "First in - First out" erledigen. Abweichungen von dieser Regel erweisen sich nur dann als erforderlich, wenn beispielsweise durch sehr hohe Druckzeiten eines Jobs die Gesamtverarbeitungszeit des Systems berührt wird.

Die Nichtbeachtung der Drucker bedingt, daß die Kernspeicherbereiche, die während des Druckens zur Verfügung gestellt werden müssen, nicht in den Bereich des für Anwenderjobs verfügbaren Kernspeichers einbezogen werden. Diese Bestimmung soll gleichermaßen für die Kartenlese- und -stanzeinheiten getroffen werden, d.h. es wird davon ausgegangen, daß das Laden von Lochkarten auf Magnetbänder nicht auf der /370-165 vorgenommen wird und daß der für das Abstanzen von Lochkarten benötigte Kernspeicherbereich den verfügbaren Kernspeicher nicht beeinträchtigt.

Nach diesen ergänzenden Feststellungen zu einzelnen Systemelementen muß nun die Frage beantwortet werden, wie die Minimierung der Gesamtverarbeitungszeit erreicht werden soll. Die Minimierung der Gesamtverar-

---

1) vgl. oben S. 163 - 164.

2) vgl. oben S. 25.

beitungszeit ist gleichbedeutend mit der Maximierung des Ausnutzungsgrades der EDV-Anlage und damit ist die Maximierung des Ausnutzungsgrades der Zentraleinheit zweifellos als ein erstes Kriterium für die Lösung des Minimierungsproblems anzusehen. Der Ausnutzungsgrad wird maximiert, wenn man rechen- und ein-/ausgabeintensive Jobs unter Beachtung der Kapazitätsgrenzen der übrigen Systemelemente so kombiniert, daß die Zentraleinheit ständig beschäftigt ist. Ihre Belegung wird im einzelnen allerdings durch das Betriebssystem gesteuert. Selbst in technisch/kaufmännisch orientierten Rechenzentren dürfte die CPU-Intensität meist im Durchschnitt nur bei 30 - 40 % liegen, so daß man eine hohe CPU-Auslastung langfristig durch verbesserte Programmiermethoden zwar anstreben sollte, kurzfristig aber kaum verwirklichen kann.

Als zweites Kriterium bietet sich die Maximierung des Belegungsgrades der peripheren Einheiten an. Da diese Aussage nicht operabel ist, muß sie noch präzisiert werden. Die Belegungsplanung muß versuchen, dasjenige Systemelement maximal auszunutzen, das von den im Jobablaufplan aufgeführten Jobs am häufigsten benötigt wird und in gewissem Sinn für einen bestimmten Tag das "Engpaßelement" darstellt. Bei fundamentaler Ungleichgewichtigkeit der verschiedenen Systemelemente in der Gesamtkonfiguration oder bei einseitiger Beanspruchung eines Systemelements durch alle Jobs wird dieser "Engpaßfaktor" auch für einen längeren Zeitraum festliegen und sich nicht verändern lassen. Bei gleichmäßiger Inanspruchnahme aller Systemelemente und einer ausgeglichenen Systemkonfiguration kann er dagegen in Abhängigkeit von den Anforderungen der Jobs, die an einem Tag erledigt werden müssen, ständig zwischen verschiedenen Systemkomponenten wechseln, d.h. an einem Tag können die Bandeinheiten, an einem anderen Tag die Plateneinheiten und an einem dritten Tag die Kernspeicher-Regions Engpaßelement sein.

Zu Beginn des Belegungsplanungsverfahrens sollten daher in jedem Planungszeitraum mindestens einmal[1] Belegungsfaktoren für die einzelnen Systemelemente bestimmt werden, die Auskunft darüber geben, in welchem Umfang die entsprechende Systemkapazität von den zu erledigenden Jobs in Anspruch genommen wird. Die Belegungsfaktoren sollen dem Ausnutzungsgrad eines Kapazitätstyps entsprechend als Verhältnis von angeforderter und zur Verfügung stehender Kapazität definiert werden. Sie zeigen dem Planungsverfahren an, welches Systemelement verstärkt bei der Einplanung zu berücksichtigen ist, um eine minimale Gesamtverarbeitungszeit zu erreichen. Der Engpaßcharakter eines Systemelements kann dabei besonders bei einer an sich ausgeglichen konfigurierten EDV-Anlage von Tag zu Tag variieren. Ergeben sich z.B. beim Vergleich der Belegungsfaktoren nur geringfügige Unterschiede, kann strenggenommen kein Systemelement als Engpaßfaktor bezeichnet werden, und man muß in diesem Fall versuchen, bei der Kombination der Jobs alle Elemente möglichst gut zu nutzen.

Das Belegungsplanungsverfahren wird auf Grund des Vergleichs der Belegungsfaktoren einen von vier Zweigen durchlaufen, die folgendermaßen gekennzeichnet werden können: 1. es liegt kein ausgesprochenes Engpaßelement vor, d.h. es ist eine insgesamt gute Inanspruchnahme der Systemkapazitäten anzustreben, 2. die verfügbaren Regions sind weitgehend zu nutzen, 3. Magnetbandeinheiten sind Engpaß und müssen möglichst ständig belegt sein und 4. der Magnetplattenspeicherplatz muß möglichst vollständig ausgenutzt werden[2]. Berechnet man die Belegungsfaktoren auch während des Einplanungsprozesses, bietet sich die Möglichkeit, innerhalb des Verfahrens andere Prioritäten für die weitere Durchführung der Belegungsplanung zu setzen und damit einen anderen Weg bei der weiteren Einplanung zu beschreiten, falls sich ein anderes Systemelement als Engpaßelement herausgestellt hat.

---

1) Natürlich kann die Berechnung der Belegungsfaktoren auch während des Einplanungsprozesses wiederholt für die dann noch nicht erledigten Jobs vorgenommen werden.

2) vgl. S. 179, Abb. 37.

Die verfügbare Kapazität eines Systemelements wird bei der Bestimmung des Ausnutzungsgrades auf die Anschaltzeit der EDV-Anlage bezogen, die nach dem Einplanungsprozeß durch den Belegungsplan ausgewiesen wird. Da diese Anschaltzeit bei der Errechnung der Belegungsfaktoren noch unbekannt ist, muß man versuchen, für die Ermittlung der verfügbaren Kapazität eines Systemelements zunächst eine vorläufige Mindestanschaltzeit A'_s der EDV-Anlage zu bestimmen, die notwendig ist, um die im Jobablaufplan angegebenen Jobs zu erledigen. Die verfügbare Kapazität ist gleich dem Produkt aus der gesuchten vorläufigen Anschaltzeit A'_s und der Anzahl der von einem Systemelement s ($s = 1,\ldots,S$) zur Verfügung stehenden Einheiten K_s. Die von einem Job i ($i = 1,\ldots,N$) angeforderte Kapazität je Systemelement ist entsprechend durch das Produkt aus der Verweilzeit VZ_i eines Jobs und der von ihm maximal benötigten Einheiten k_{is} des Systemelements festgelegt. Summiert man die von den einzelnen Jobs je Systemelement angeforderten Kapazitäten auf, erhält man die von allen Jobs eines Tages insgesamt benötigte Kapazität. Der Belegungsfaktor BF_s eines Systemelements ist dann gleich dem Quotienten aus der insgesamt angeforderten Kapazität K_a und der verfügbaren Kapazität K_v:

$$BF_s = \frac{K_a}{K_v} = \frac{\sum_{i=1}^{N} k_{is} \cdot VZ_i}{K_s \cdot A'_s} \quad , \; (i = 1,\ldots,N; \quad s = 1,\ldots,S)$$

Da in dieser Gleichung neben der vorläufigen Mindestanschaltzeit A' auch der Belegungsfaktor BF_s unbekannt ist, kann man die gesuchte Anschaltzeit A'_s nur bestimmen, wenn der Belegungsfaktor eines beliebigen Systemelements vorgegeben wird und die Belegungsfaktoren der übrigen Systemelemente dann auf Grund der so ermittelten Anschaltzeit A'_s berechnet werden. Diese Vorgabe ist zulässig, weil die Belegungsfaktoren nur die relative Knappheit der Systemelemente bei einer vorgegebenen Anzahl von Jobs widerspiegeln sollen.

Legt man beispielsweise fest, daß der Belegungsfaktor für die Kernspeicher-Regions BF_s (s = 1) gleich 1 ist[1], so läßt sich durch eine einfache Umformung die im Hinblick auf die Verfügbarkeit der Regions und die Regionanforderungen durch die Jobs eines Tages mindestens erforderliche Anschaltzeit

nach der Beziehung $A'_1 = \frac{\sum_{i=1}^{N} k_{i1} \cdot VZ_i}{K_1 \cdot BF_1}$ errechnen. Die Anzahl

K_1 der Jobs, die in der Multiprogrammverarbeitung auf Grund des verfügbaren Kernspeicherplatzes parallel zueinander ablaufen können, läßt sich besonders einfach bestimmen, wenn bei der Systeminitialisierung von der Möglichkeit zur Festlegung der Größe der Kernspeicher-Regions Gebrauch gemacht wurde[2]. In diesem Fall gibt k_1 die Anzahl der von einem Job angeforderten Regions und K_1 die Zahl der für den Parallellauf verfügbaren Regions an. Bei variabler Regiongröße müssen sowohl K_1 als auch k_1 in Kilo-Bytes vorgegeben werden.

Die Berechnung der Belegungsfaktoren für die Magnetbandeinheiten (BF_s, s = 2) und die Plattenspeichereinheiten (BF_s, s = 3) kann auf Grund der festgelegten Anschaltzeit A'_1 vorgenommen werden. So ist beispielsweise

$BF_2 = \frac{\sum_{i=1}^{N} k_{i2} \cdot VZ_i}{K_2 \cdot A'_1}$, wobei k_{i2} die Anzahl der von einem Job angeforderten Bandeinheiten, VZ_i seine Verweilzeit, K_2 die insgesamt zur Verfügung stehenden Bandeinheiten und A'_1 die Anschaltzeit darstellt, die ohne Berücksichtigung der Jobabhängigkeiten für einen Belegungsfaktor der Kernspeicher-Regions $BF_1 = 1$ ermittelt wurde. Für die Magnetplattenspeichereinheiten wird die Berechnung entsprechend durchgeführt, wobei bei der Bestimmung aller Belegungsfaktoren stets ausgeschlossen sein muß, daß ein Job mehr Einheiten eines Systemelements anfordert als insgesamt vorhanden sind.

---

1) Diese Festlegung empfiehlt sich, weil jeder Job mindestens eine Region belegt und der Zähler des angegebenen Ausdrucks dadurch nicht Null werden kann.

2) vgl. oben S. 162.

Die Belegungsfaktoren können theoretisch alle Werte annehmen, die ≥ 0 sind. Der Wert Null ergibt sich, wenn das entsprechende Systemelement von keinem Job benötigt wird, der Wert 1 stellt die Schwelle dar, der Auskunft gibt, ob die zunächst nur vorläufig festgelegte Anschaltzeit A'_s noch korrigiert werden muß. Stellt sich beispielsweise heraus, daß der Belegungsfaktor für Bandeinheiten den Wert $BF_2 = 1,25$ und der Belegungsfaktor für Plattenspeichereinheiten den Wert $BF_3 = 0,85$ annimmt, können die im Jobablaufplan angegebenen Jobs nicht in der Zeit A'_1 erledigt werden, sondern sie benötigen zu ihrer Durchführung mindestens die Zeit $A^* = A'_1 \cdot BF_{max}$, wobei BF_{max} hier gleich dem Belegungsfaktor für Bandeinheiten wäre. Während die mindestens erforderliche Anschaltzeit also neu berechnet werden muß, wenn ein Belegungsfaktor einen Wert ergibt, der größer als 1 ist, erübrigt sich diese Neuberechnung für die Belegungsfaktoren, da sie nur die relative Knappheit eines Systemelements ausdrücken sollen, obwohl sie an sich - analog zum Ausnutzungsgrad - auch höchstens den Wert 1 erreichen dürften. Die so ermittelte minimale Anschaltzeit A^* muß nun allerdings noch mit der Gesamtdauer des zeitlängsten Jobblocks verglichen und eventuell nochmals korrigiert werden[1]. Ist die höchste Blockdauer größer als der für A^* berechnete Wert, muß die Dauer des zeitlängsten Blocks als minimal erreichbare Anschaltzeit angesehen werden, da die Abarbeitung des entsprechenden Blocks dazu zwingt, die EDV-Anlage mindestens für diese Dauer anzuschalten.

Die minimale Anschaltzeit A^* dürfte durch den Belegungsplan fast immer überschritten werden, weil bei ihrer Ermittlung die Jobabhängigkeiten - bis auf den soeben geschilderten Fall - nicht beachtet werden. An Hand dieser Größe ist aber eine Beurteilung des Ergebnisses der Belegungsplanung möglich. Die Abweichungen bei verschiedenen vom Verfasser durchgeführten Testläufen zwischen der durch das Planungsverfahren erhaltenen und der minimal erreichbaren Anschaltzeit A^* lagen bei etwa 10 %.

---

1) Zur Berechnung der einzelnen Jobblockdauern muß zu Beginn des Belegungsplanungs-Verfahrens die in Kapitel 4 erläuterte Zeitrechnung durchgeführt werden (vgl. oben S. 94).

Der größte Belegungsfaktor weist auf den Kapazitätstyp hin, der für die Jobs einer Planungsperiode als Engpaßfaktor anzusehen und daher vom Planungsverfahren mit höherer Priorität als die übrigen Systemelemente zu behandeln ist, es sei denn, die Differenz zu den anderen Faktoren ist so gering, daß kein Systemelement als Engpaßfaktor zu betrachten ist.

Die Verfügbarkeit der einzelnen Systemelemente wird dem Planungsverfahren über jeweils eine Parameterkarte pro Systemelement vorgegeben. Je nach der Länge des Zeitraums, für den die Belegungsplanung erfolgen soll, genügt dabei entweder eine konstante Vorgabe, oder es wird die Möglichkeit vorgesehen, im Zeitablauf variable Kapazitätsgrenzen zu berücksichtigen, indem angegeben wird, in welcher Höhe die Kapazität bis zu bestimmten Zeitpunkten zur Verfügung steht. Bei einer Änderung der Kapazitätsverfügbarkeit müssen die Belegungsfaktoren in jedem Fall neu berechnet werden.

Das Belegungsplanungsverfahren arbeitet parallel[1], d.h. es sucht zu einem bestimmten Zeitpunkt des Einplanungsprozesses aus allen noch zu erledigenden Jobs diejenigen heraus, die zu diesem Zeitpunkt einplanbar sind und versucht sie unter Beachtung der Kapazitätsgrenzen einzuplanen. Dieses parallele Verfahren erfordert einen Zeitgeber, der die einzelnen Einplanungszeitpunkte repräsentiert. Da die Schätzung der Druck- und Verweilzeiten immer nur auf volle 5 oder 10 Minutenabstände angegeben wird, soll - vor allem aus speicherungstechnischen Gründen - ein Schritt des Zeitgebers einem 5-Minuten-Intervall entsprechen. Ein Job, der 45 Minuten Verweilzeit hat, wird entsprechend in neun Zeitgeberintervallen eingeplant. Die Jobstruktur sowie die für die Belegungsplanung benötigten Jobinformationen erhält das Planungsverfahren aus dem Programm zur Jobablaufplanung[2]. Benötigt werden: der Jobname, die Job-Blocknummer, der Rang des Jobs, die Verweilzeit und die Druckzeit,

---

1) vgl. oben S. 139.

2) vgl. oben S. 100.

der Bereitstellungstermin, der Abgabetermin, die Kapazitätsanforderungen je Systemelement, die Jobpriorität, die Jobintensität (CPU-Zeit/Verweilzeit), Folgejobangaben sowie Angaben über Jobs, die gleiche Eingabefiles anfordern.

Aus dieser Aufzählung geht hervor, daß das Planungsverfahren, das nun im einzelnen erläutert werden soll, in zwei Fällen eine bevorzugte Einplanung von Jobs vornehmen muß, nämlich einmal beim Vorhandensein einer Jobpriorität ("PRIORITY"-Angabe in der Job-Karte) und zum anderen, wenn ein Abgabetermin beachtet werden muß. Insgesamt muß das Verfahren also dreistufig arbeiten, wobei auf der zweiten und der dritten Stufe noch eine Untergliederung in weitere Einplanungskategorien bzw. Planungszweige zu beachten ist. Auf der ersten Stufe sind alle Jobs mit absolutem Vorrang einzuplanen, die eine PRIORITY-Angabe enthalten; beim IBM-Betriebssystem können hier graduell verschiedene Prioritäten vorgegeben werden. In der zweiten Stufe ist den Jobs besondere Beachtung zu schenken, die einen Abgabetermin einzuhalten haben,und erst auf der dritten Stufe kann die Einplanung der "normalen" Jobs vorgenommen werden.

Zu Stufe 1:

Jobs mit einer PRIORITY-Angabe, die über der Prioritätsziffer anderer Jobs liegt, müssen vom Planungsverfahren vorrangig eingeplant werden, weil auch die Routinen des Betriebssystems diesen Jobs Kapazitäten zuordnen, sobald sie zur Verfügung stehen. Bei der Belegungsplanung wird außerdem davon ausgegangen, daß nicht nur der Job bevorzugt eingeplant werden muß, der die höhere Prioritätsziffer enthält, sondern auch die Jobs, die in seinem Block bzw. seiner Jobfolge einen um mindestens 1 geringeren Rang als er selbst besitzen. Dieses Vorgehen ergibt sich zwar nicht zwingend vom Betriebssystem her, aber man kann davon ausgehen, daß auch den logisch vor ihm zu erledigenden Jobs eine erhöhte Priorität zukommt. Eine Analyse der einzelnen Blöcke erlaubt die entsprechende Prioritätsvorgabe. Ist bei den Jobs mit höherer Priorität zu-

sätzlich eine Angabe über den Bereitstellungstermin gemacht, wird dieser selbstverständlich beachtet, d.h. der Job wird erst als einplanbar angesehen, wenn der Bereitstellungstermin erreicht ist; Abgabetermine können dagegen unberücksichtigt bleiben, da ja gerade durch die höhere Prioritätsziffer sichergestellt ist, daß der Job sofort eingeplant wird. Bedingung für die sofortige Einplanung ist allerdings, daß der Job entweder keine Vorjobs besitzt, d.h. im Rang 0 steht oder daß seine Vorjobs erledigt sind. Da bei dem Betriebssystem OS-MVT die Prioritätszifffern von 1 - 14 vergeben werden können, erfolgt natürlich auch die Einplanung der Höhe der Prioritätsziffer entsprechend.

Zu Stufe 2:

Während bei der Jobablaufplanung, in der es lediglich um die Ermittlung der logischen Jobabhängigkeiten ging, Abgabetermine nicht beachtet zu werden brauchten, ist ihre Berücksichtigung im Rahmen der Belegungsplanung zwingend erforderlich, weil das Planungsverfahren den Start- und Endzeitpunkt jedes Jobs genau fixiert. Abgabetermine treten erfahrungsgemäß vor allem am Ende eines Jobblocks auf und geben den spätesten Termin an, zu dem ein Job erledigt sein muß. Aus der Definition des Abgabetermins in Kapitel 3.2 [1] geht hervor, daß die Erledigung des Jobs sowohl die eigentliche Verarbeitungszeit (Verweilzeit) als auch die Druckzeit umschließt. Der späteste Anfangstermin, der für die Belegungsplanung maßgebend ist, ergibt sich also aus Abgabetermin ./. Verweilzeit ./. Druckzeit. Da die Reihenfolge für das Drucken der Ausgabelisten durch das Belegungsplanungsverfahren nicht vorgegeben wird, hat es sich als günstig erwiesen, Jobs mit einem Abgabetermin im Belegungsplan besonders zu kennzeichnen. Sind die Drucker durch die Ausgabelisten anderer - terminlich nicht gebundener - Jobs noch für längere Zeit belegt, kann dann von einem Operator die normale Ausgabereihe "First in - First out" unterbrochen werden und der Ausgabe des entsprechenden Jobs eine höhere Priorität zugeordnet werden, die sicherstellt, daß seine Ausgabelisten als nächste gedruckt werden.

---

1) vgl. oben S. 58.

Besonders zu beachten ist bei den Jobs mit Abgabeterminen, daß auch ihren Vorjobs "fiktive" Abgabetermine bzw. späteste Anfangstermine zugewiesen werden müssen, um die termingerechte Fertigstellung des eigentlich interessierenden Jobs sicherzustellen. Die Berücksichtigung der Druckzeit erübrigt sich allerdings bei der Festlegung der fiktiven Termine, da die Druckausgabelisten dieser Jobs ja nicht bis zu einem bestimmten Zeitpunkt vorliegen müssen. Ihre spätesten Anfangstermine werden festgestellt, indem man - ähnlich wie bei der Rückwärtsrechnung[1] - vom kleinsten spätesten Anfangstermin eines ihrer Folgejobs die eigene Verweilzeit subtrahiert. Treten in einem Jobblock mehrere Jobs mit Abgabeterminen in verschiedenen Rängen auf, muß die Druckzeit selbstverständlich bei jedem Job mit Abgabetermin berücksichtigt werden. Als spätesten Anfangstermin eines Jobs, dem ein fiktiver Abgabetermin zugeordnet werden würde, weil einer seiner Folgejobs einen Abgabetermin besitzt, der aber selbst einen Abgabetermin enthält, wählt man den zeitlich früher liegenden Wert aus und rechnet mit ihm weiter. Besitzt ein Vorjob zusätzlich einen Bereitstellungstermin[2], so muß auch dieser vom Einplanungsverfahren berücksichtigt werden. Ist die Differenz zwischen dem spätesten Anfangs- und dem Bereitstellungstermin dieses Jobs negativ, d.h. müßte dieser Job nach dem Ergebnis der Zeitrechnung bereits gestartet werden, bevor die Daten für ihn bereitgestellt werden können, muß vom Planungsverfahren eine Meldung erfolgen, die auf die fehlerhaften Angaben hinweist und den Abgabetermin des entsprechenden Jobs in Frage stellt.

Sowohl Jobs mit einer erhöhten Jobpriorität als auch Jobs mit Abgabeterminen können die Güte des Belegungsplans sehr stark beeinträchtigen, da die Kombination dieser Jobs nicht im Hinblick auf eine möglichst weitgehende Belegung des Engpaßelements vorgenommen werden kann. Während

---

1) vgl. oben S. 96.

2) Während der Bereitstellungstermin angibt, wann ein Job frühestens zur Ausführung eingeplant werden darf, legt der Abgabetermin fest, wann ein Job spätestens - einschließlich der Erstellung der Drucklisten - beendet sein muß.

Jobprioritäten unbedingt beachtet werden müssen, kann man die Beeinträchtigung des Plans durch die Abgabetermine zumindest teilweise umgehen, indem man diese Jobs bereits vor der Erreichung ihres spätesten Anfangstermins im Rahmen der normalen Einplanung berücksichtigt. Dadurch läßt sich der Belegungsplan häufig nicht unwesentlich verbessern. Die Dringlichkeit der Jobs mit Abgabeterminen wird vom Belegungsplanungsverfahren aus diesem Grunde in drei Kategorien aufgeteilt, die erkennen lassen, ob ein solcher Job im Hinblick auf seinen Abgabetermin kritisch, subkritisch oder nicht kritisch ist.

In die erste Kategorie der kritischen Jobs fallen die Jobs, bei denen die Differenz zwischen spätestem Anfangstermin und umgerechnetem augenblicklichen Stand des Zeitgebers kleiner als 30 Minuten ist. Sie sollen sofort eingeplant werden, wenn entsprechende Kapazitäten frei sind. Die Differenz von 30 Minuten wurde gewählt, weil sich bei Untersuchungen herausgestellt hat, daß im Durchschnitt alle 5 - 10 Minuten ein Job endet. Steht beim erstenmal, wenn der Job in diese Kategorie fällt, nicht genügend Peripherie zur Verfügung, ist auf jeden Fall anzunehmen, daß der Job beim nächsten oder übernächsten Schritt des Zeitgebers eingeplant werden kann. Man umgeht mit dieser Methode die Notwendigkeit, einen bereits normal eingeplanten Job wieder aus dem Belegungsplan zu streichen, um die dadurch freigewordene Kapazität einem Job mit Abgabetermin zuweisen zu können. Ein solches Vorgehen würde nämlich eine bedeutende Verlangsamung des Planungsverfahrens zur Folge haben.

In die zweite Kategorie fallen die subkritischen Jobs, bei denen die Differenz zwischen spätestem Anfangstermin und dem jeweiligen Zeitpunkt des Einplanungsprozesses zwischen 30 und 60 Minuten liegt. Diese Jobs werden der dritten Stufe des Planungsverfahrens zugewiesen, die die Einplanung der Jobs vornimmt, die nicht auf Grund einer "PRIORITY"-Angabe oder eines Abgabetermins eine höhere Priorität besitzen. Durch die Zuordnung dieser Jobs zur dritten Stufe des Planungsverfahrens soll versucht werden, durch ihre frühzeitige Einplanung eine günstigere Belegung der Systemelemente und damit eine kürzere Gesamt-

verarbeitungszeit zu erreichen als dies möglich ist, wenn die Jobs hinsichtlich des Abgabetermins kritisch sind und ohne Rücksicht auf die Ausnutzung der Systemelemente eingeplant werden müssen.

Die dritte Kategorie umfaßt schließlich die nicht kritischen Jobs mit Abgabetermin, die einen Spielraum von mehr als 60 Minuten bis zum spätesten Anfangstermin besitzen. Diese Jobs werden ebenfalls wie normale Jobs behandelt und das heißt, sie können beim Ablauf der dritten Stufe des Planungsverfahrens in den Belegungsplan gelangen[1]. Tests mit entsprechenden Jobs haben ergeben, daß durch die Bildung dieser Kategorien etwa 60 - 80 % der Jobs mit einem Abgabetermin bereits eingeplant waren, bevor sie kritisch wurden.

Zu Stufe 3:

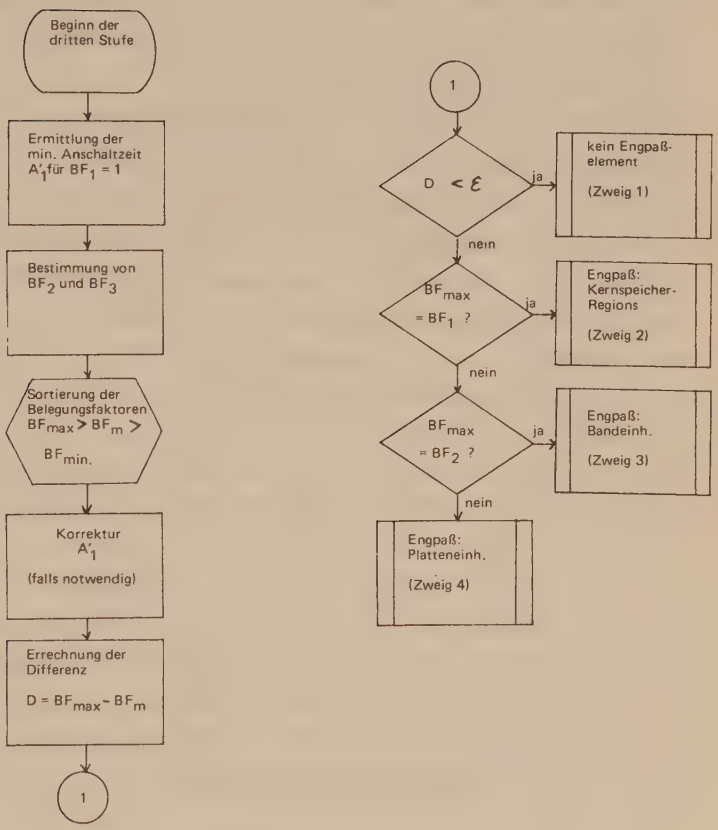
In der dritten und wichtigsten Stufe des Planungsverfahrens werden die Jobs betrachtet, die weder eine höhere Jobpriorität besitzen noch einen Abgabetermin einzuhalten haben (Anteil an der Gesamtzahl der einzuplanenden Jobs ca. 90 %).

Da die Minimierung der Gesamtverarbeitungszeit durch die Maximierung des Ausnutzungsgrades des Engpaßelements erfolgen soll, wird die Einplanung der Jobs auf der dritten Stufe so vorgenommen, daß das Engpaßelement möglichst vollständig und die übrigen Systemelemente weitgehend belegt werden. Wie das vereinfachte Ablaufdiagramm in Abb. 37 zeigt, erfolgt die Bestimmung des Engpaßelements und die Auswahl des für die Belegungsplanung günstigsten Zweiges auf der dritten Stufe automatisch durch das Einplanungsverfahren.

---

1) Die genaue Behandlung der subkritischen und der nicht kritischen Jobs in der dritten Stufe des Planungsverfahrens wird noch im einzelnen erläutert.

Abb. 37 : Ablaufdiagramm für die automatische Festlegung des Engpaßelements und die Auswahl des für die Belegungsplanung günstigsten Zweiges



Bereits oben wurde erläutert, daß das Planungsverfahren in Abhängigkeit vom Engpaßelement einen von vier Zweigen durchlaufen muß [1]. Um den günstigsten Zweig automatisch zu finden, ordnet man die Belegungsfaktoren in absteigender Folge, so daß der maximale Belegungsfaktor BF_{max} an erster Stelle, der Faktor BF_m mit dem zweitgrößten Wert an zweiter und der

1) vgl. oben S. 169.

kleinste Belegungsfaktor BF_{min} an letzter Stelle steht. Dann wird die Differenz $D = BF_{max} - BF_{m}$ zwischen dem größten und dem zweitgrößten Belegungsfaktor gebildet und geprüft, ob diese Differenz D kleiner ist als ein vorgegebener Wert ε [1]. Ist das der Fall, wird kein Systemelement als Engpaßelement angesehen. Das Planungsverfahren versucht dann alle Systemelemente weitestgehend auszunutzen (Zweig 1), indem es jeweils feststellt, welches Systemelement bei dem gegenwärtigen Stand des Einplanungsprozesses in Relation zu den anderen Elementen am wenigsten belegt ist. Dieses Systemelement wird für den betrachteten Zeitpunkt als Engpaßelement betrachtet und auf einem der Zweige 2, 3 oder 4 so weit wie möglich belegt. Zweig 2 würde eingeschlagen, wenn die Kernspeicher-Regions besser belegt werden müßten, Zweig 3 bei einer schlechten Nutzung der Magnetbandeinheiten und Zweig 4, wenn die Belegung der Plattenspeichereinheiten verstärkt werden soll. Zweig 1 ist zwar sehr kurz, beinhaltet aber den aufwendigsten Teil des Planungsverfahrens, weil nach seiner Ausführung stets noch ein anderer Zweig, nämlich Zweig 2, 3 oder 4, zusätzlich durchlaufen werden muß.

Ist das Engpaßelement hingegen zum Beginn des Einplanungsverfahrens bereits eindeutig bestimmbar ($D \geq \varepsilon$), wird ein anderer Zweig des Verfahrens nur beschritten, wenn sich bei einer Neuberechnung der Belegungsfaktoren für die noch nicht eingeplanten Jobs ein anderes Engpaßelement ergibt. Diese Neuberechnung sollte nach Erfahrungen des Verfassers erfolgen, wenn Jobs mit einer Gesamtverarbeitungszeit eingeplant sind, die einer Anschaltzeit der EDV-Anlage von etwa zwei Stunden entspricht[2].

Die Zweige 2, 3 und 4 des Planungsverfahrens sind vom Aufbau her gesehen sehr ähnlich, da jeweils nur ein anderes Systemelement mit höherer Priorität einzuplanen ist. Die Unterteilung in verschiedene Zweige ist hauptsächlich aus programmiertechnischen Überlegungen erfolgt, um eine größt-

---

1) Bei Tests mit dem Belegungsplanungsverfahren haben sich mit einem Wert $\varepsilon = 0{,}10$ sehr gute Ergebnisse erzielen lassen.

2) Dies ist nach jeweils 25 Schritten des Zeitgebers der Fall.

mögliche Schnelligkeit in der Ausführung des Verfahrens zu erreichen. Da nach der ausführlichen Darstellung eines Zweiges die Beschreibung der übrigen Zweige nur wiederholenden Charakter besäße, soll im folgenden nur der Zweig 3 näher erläutert werden, in dem die Bandeinheiten als Engpaß angesehen werden[1].

Aus der Definition der Mindestanschaltzeit A^* folgt, daß die minimal erforderliche Gesamtverarbeitungszeit nur erreicht werden kann, wenn das Planungsverfahren das Engpaßelement in jedem Zeitpunkt des Belegungsplans vollständig in Anspruch nimmt. Gelingt die vollständige Belegung nicht, ist mit Sicherheit eine Belegungsdauer zu erwarten, die über der minimalen Anschaltzeit liegt. Um eine minimale Belegungsdauer zu erreichen, arbeitet das Planungsverfahren durchsatzorientiert, d.h. es versucht, zu einem gegebenen Zeitpunkt immer eine möglischt große Anzahl von Jobs einzuplanen. Bei fest vorgegebener Regiongröße läßt sich die max. Anzahl der parallel einzuplanenden Jobs an Hand des verfügbaren Kernspeicherplatzes unschwer ermitteln, bei variabler Regiongröße ist sie dem Verfahren als Planungsparameter vorzugeben. Die Aussage, daß das Verfahren durchsatzorientiert arbeitet, bedeutet nicht, daß die Kernspeicherausnutzung in den Mittelpunkt gestellt wird, sondern sie gibt lediglich einen Hinweis für die Auswahl und Kombination der Jobs[2]. Die Einplanung möglichst vieler Jobs unter Beachtung der verfügbaren Kapazitäten ist nur durchführbar, wenn das Verfahren zunächst die Jobs auswählt, die das Engpaßelement, in diesem Falle also die Bandeinheiten, am

---

1) Das Planungsverfahren wurde vom Verfasser für das Rechenzentrum entwickelt, dessen Konfiguration in Kap. 6.2 erläutert wurde. In diesem Rechenzentrum ist die Region-Größe auf 104 KB festgelegt, obwohl die MVT-Version des Operating Systems der IBM benutzt wird. Der verfügbare Kernspeicherplatz läßt sich durch diese Festlegung sehr leicht in die Anzahl der Jobs umrechnen, die maximal in einem Zeitpunkt eingeplant werden können. Bei der Erläuterung des Verfahrens wird von dieser Festlegung ausgegangen.

2) Bei fest vorgegebener Regiongröße erfolgt allerdings gleichzeitig neben der möglichst vollständigen Nutzung des Engpaßelements auch eine gute Ausnutzung des Kernspeichers.

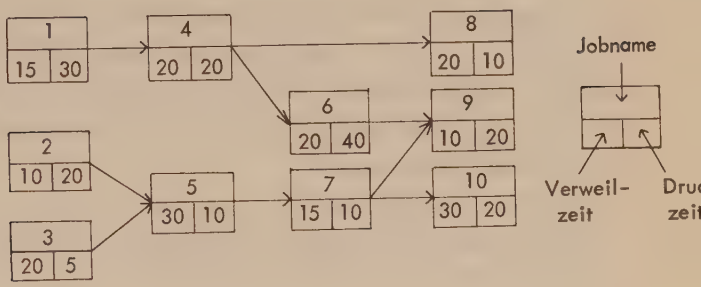
wenigsten belasten. Können beispielsweise drei Jobs zusätzlich eingeplant werden, wählt man aus den einplanbaren Jobs zunächst die drei Jobs aus, die die Magnetbandeinheiten im geringsten Maße in Anspruch nehmen und überprüft dann erst in einem zweiten Schritt, ob sie auf Grund der verfügbaren Bandeinheiten eventuell durch Jobs mit einer höheren Anforderung ersetzt werden können. Eine endgültige Einplanung der Jobs erfolgt erst, wenn in bezug auf das Engpaßelement keine bessere Belegung gefunden wird.

Häufig benötigen mehr Jobs die gleiche Anzahl von Bandeinheiten als zusätzliche Jobs in den Belegungsplan aufgenommen werden können. Die Auswahl der einzuplanenden Jobs sollte dann keinesfalls dem Zufall überlassen werden. Vielmehr bietet sich hier die Möglichkeit, durch die Einführung eines weiteren Auswahlkriteriums verstärkt auf eine Minimierung der Gesamtverarbeitungszeit hinzuarbeiten.

Bedenkt man, daß bei der minimalen Anschaltzeit A^* das Engpaßelement bis zum letzten Zeitpunkt des Belegungsplans voll genutzt sein muß, folgt daraus, daß die Jobkombination möglichst so erfolgen muß, daß am Ende der Belegungsplanung keine Jobs mehr eingeplant werden, die erstens eine besonders hohe Druckzeit erfordern, die zweitens eine wesentlich höhere Verweilzeit als die übrigen noch einzuplanenden Jobs haben oder drittens selbst noch mehrere Folgejobs freigeben.

Diese Einschränkungen berücksichtigt eine Größe, die als Restzeit eines Jobs bezeichnet werden soll. Ihre Bestimmung erfolgt im Anschluß an die Zeitrechnung, die für jeden Jobblock gesondert durchzuführen ist. Und zwar berechnet man für jeden Job die Summe aus seiner Verweilzeit und seiner Druckzeit und vergleicht diesen Wert mit der Differenz, die aus dem spätesten Endzeitpunkt des zugehörigen Blocks und dem spätesten Anfangszeitpunkt des betrachteten Jobs gebildet wird. Der größere dieser beiden Werte gibt die Restzeit eines Jobs an. Für den Jobblock in Abb. 38 soll die Ermittlung der Restzeiten in Tab. 26 gezeigt werden.

Abb. 38 : Jobblock zur Bestimmung der Restzeiten



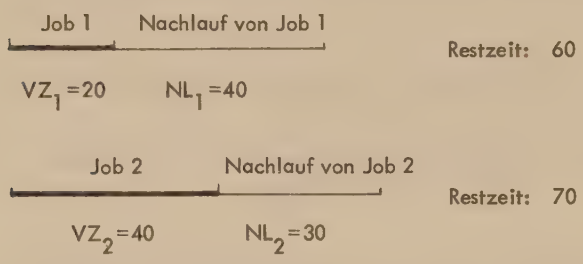
Tab. 26 : Zeitrechnung und Bestimmung der Restzeiten für Abb. 38

Jobname	Verweilzeit	Frühester Anfangs-zeitpunkt (FAZ)	Frühester End-zeitpunkt (FEZ)	Spätester Anfangs-zeitpunkt (SAZ)	Spätester End-zeitpunkt (SEZ)	Druckzeit	Restzeit 1	Restzeit 2	gültige Restzeit
	1	2	3	4	5	6	1 + 6	95 - 4	
1	15	0	15	30	45	30	45	65	65
2	10	0	10	10	20	20	30	85	85
3	20	0	20	0	20	5	25	95	95
4	20	15	35	45	65	20	40	50	50
5	30	20	50	20	50	10	40	75	75
6	20	35	55	65	85	40	60	30	60
7	15	50	65	50	65	10	25	45	45
8	20	35	55	75	95	10	30	20	30
9	10	65	75	85	95	20	30	10	30
10	30	65	95	65	95	20	50	30	50

Das Planungsverfahren wählt von den Jobs, die die gleiche Anzahl von Bandeinheiten benötigen, immer denjenigen zuerst aus, der die größte Restzeit besitzt. Dadurch wird erreicht, daß in der Endphase der Belegungsplanung nur noch Jobs eingeplant werden müssen, die eine kleine Restzeit haben, also am ehesten eine minimale Gesamtverarbeitungszeit versprechen. Dieses Vorgehen ist in gewisser Weise mit dem bei der Variationen-Enumeration

verwendeten Auswahlprinzip des maximalen Nachlaufs[1] vergleichbar. Es bietet die zusätzlichen Vorteile, sowohl die Druckzeiten der Jobs in die Planungsüberlegungen einzubeziehen als auch ihre Verweilzeiten zu berücksichtigen. Wie Abb. 39 zeigt, genügt die Bestimmung des Nachlaufs nicht als Auswahlkriterium, da bei der Belegungsplanung auf Grund des parallelen Aufbaus der Jobkombinationen immer der auf den Start bezogene Zeitraum betrachtet werden muß.

Abb. 39 : Beispiel zur Erläuterung des Unterschiedes zwischen dem Nachlauf und der Restzeit eines Jobs



Job 1 hat in Abb. 39 zwar einen größeren Nachlauf als Job 2, d.h. nach seinem Ende vergeht bis zum Blockende die größere Zeitspanne, die Einplanung muß jedoch bereits beim möglichen Startzeitpunkt entscheiden können, welcher Job zur insgesamt längeren Belegung des Rechners führt. Das ist in diesem Fall Job 2.

Das Auswahlkriterium der maximalen Restzeit erlaubt gleichzeitig die sinnvolle Einplanung von Jobs mit gleichen Eingabedateien. Da diese Jobs nicht parallel zueinander eingeplant werden dürfen[2], obwohl dies von den logischen Abhängigkeiten her gesehen möglich wäre, muß ihre Restzeit korrigiert werden. Es bietet sich an, zur Restzeit eines solchen Jobs die Verweilzeiten der Jobs zu addieren, die die gleichen Eingabedateien benötigen wie er selbst. Durch die höhere Restzeit kann der

---

1) vgl. oben S. 133.

2) vgl. oben S. 103.

sequentiell erfolgenden Einplanung dieser Jobs am besten Rechnung getragen werden. Von mehreren einplanbaren Jobs plant man immer den mit der höchsten Restzeit ein, wobei nach der festen Einplanung eines Jobs die frühestmöglichen Anfangszeitpunkte der übrigen Jobs dann entsprechend korrigiert und die Verweilzeit dieses Jobs von den Restzeiten der Jobs mit den gleichen Eingabedateien subtrahiert werden muß.

In einem bestimmten Zeitpunkt verfügt das Planungsverfahren über i_t Jobs mit der geringsten Anforderung an Bandeinheiten und der jeweils größten Restzeit, wobei i_t hier durch die Anzahl der verfügbaren Regions bestimmt wurde[1]. In diese zeitpunktbezogene Vorplanung gelangen allerdings auch die subkritischen und die nicht kritischen Jobs mit Abgabetermin aus der zweiten Stufe des Verfahrens, so daß nicht immer die Jobs mit der geringsten Bandeinheitenanforderung ausgewählt werden. Analog zu Tab. 27, in der die 27 möglichen Ausgangsbedingungen aufgeführt sind, die nach der Vorplanung gegeben sein können, stellt das Planungsverfahren automatisch fest, ob die vorgeplanten Jobs fest eingeplant werden können, ob eine Verbesserung der Belegung möglich ist oder ob bereits durch die Vorplanung Kapazitätsgrenzen überschritten wurden und deshalb Jobs gelöscht werden müssen.

Die Tab. 27 ist in Form einer Entscheidungstabelle aufgebaut, in der auf gewisse Ausgangsbedingungen bestimmte Maßnahmen folgen. Auf der Seite der Ausgangsbedingungen wird hier jeweils die Differenz aus der Belegung der Systemelemente und ihrer Verfügbarkeit gebildet, auf der Maßnahmenseite wird dargestellt, ob die sofortige Einplanung der Jobs, die Einplanung nach Streichung eines oder mehrerer Jobs, eine eventuelle Steigerung der Bandeinheitenbelegung oder eine Streichung mit anschließender Erhöhung der Bandeinheitenbelegung in Frage kommt. Vom Ergebnis der Streichung im Hinblick auf die Belegung der Bandeinheiten hängt dabei jeweils ab, ob sofort eingeplant oder vor der Einplanung noch eine Belegungssteigerung versucht wird. Sofort eingeplant wird nur, wenn die Bandein-

1) vgl oben S. 181, Fußnote 1.

Tab. 27 Darstellung der möglichen Ausgangsbedingungen und der erforderlichen Maßnahmen in der dritten Stufe des Planungsverfahrens nach der Auswahl von j_t Jobs mit der kleinsten Bandeinheitenbelegung und der größten Restzeit (j_t = Anzahl der zum Zeitpunkt t zusätzlich einzuplanenden Jobs)

Laufende Nummer	Ausgangsbedingungen									Maßnahmen				
	Bandeinheitenbelegung B_b abzüglich Bandeinheiten verfügbarkeit B_v			Kernspeicherbelegung K_b abzüglich Kernspeicherverfügbarkeit K_v			Plattenspeicherbelegung P_b abzüglich Plattenspeicherverfügbarkeit P_v			Einplanen [1]	Bandeinheitenbelegung erhöhen [1]	Job streichen im Hinblick auf		
	< 0	= 0	> 0	< 0	= 0	> 0	< 0	= 0	> 0			Bandeinheitenbelegung	Kernspeicherbelegung	Plattenspeicherbelegung
1	x			x			x			x				
2	x			x				x		x				
3	x			x					x	x				x
4	x				x			x			x			
5	x				x				x		x			
6	x				x						x			
7	x					x			x		x			x
8	x					x					x		x	
9	x					x		x			x		x	
10		x		x					x	x				x
11		x		x				x		x				
12		x		x					x	x	x			
13		x			x			x		x				x
14		x			x				x	x				
15		x			x					x	x			
16		x				x		x		x	x			
17		x				x			x	x	x		x	
18		x				x				x	x		x	
19			x	x				x		x	x	x		
20			x	x					x	x	x	x		
21			x	x						x	x	x		x
22			x		x			x		x	x	x		
23			x		x				x	x	x	x		
24			x		x					x	x	x		x
25			x			x	x			x	x	x	x	
26			x			x		x		x	x	x	x	
27			x			x			x	x	x	x	x	x

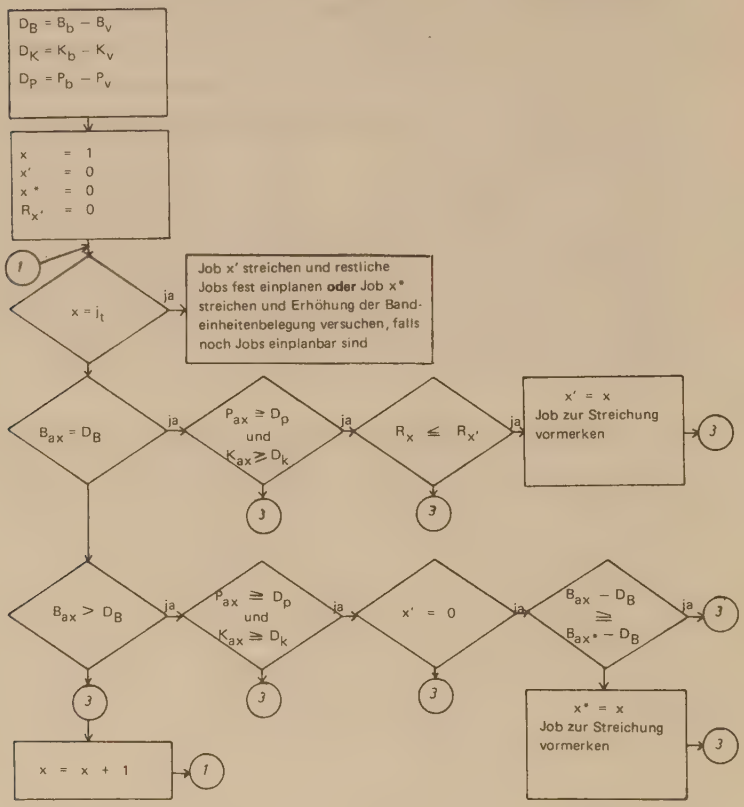
1) In den Fällen 12 und 15-27 erfolgt die sofortige Einplanung nur, wenn Jobs gefunden werden, deren Streichung zur vollständigen Belegung der Bandeinheiten führt. Anderenfalls wird versucht, die Belegung der Bandeinheiten noch zu verbessern.

heiten in vollem Umfang genutzt sind und die Verfügbarkeit bei keinem anderen Systemelement überschritten ist. In den Zeilen 1 und 2 erfolgt die Einplanung, weil außer den vorgeplanten keine weiteren Jobs zur Einplanung zur Verfügung stehen, da sonst die verfügbaren Regions voll ausgenutzt worden wären, und in den Zeilen 10, 11, 13 und 14 führt die Vorplanung genau zur vollständigen Belegung der Bandeinheiten, so daß auch hier im Hinblick auf das Engpaßelement keine Verbesserung möglich ist. In den Zeilen 4 und 5 kann schließlich sofort nach einer verbesserten Auslastung gesucht werden, weil die verfügbaren Regions zwar voll belegt, die Bandeinheiten aber noch nicht in vollem Umfang genutzt werden. Eine höhere Belegung ist allerdings nur zu erreichen, wenn noch weitere einplanbare Jobs zur Auswahl stehen.

In allen anderen Zeilen muß zunächst die Streichung eines oder mehrerer Jobs durchgeführt werden, weil die Verfügbarkeit von Systemelementen überschritten wurde. Die Streichung erfolgt entweder unter der Voraussetzung, daß die Bandeinheiten durch die Vorplanung nicht bzw. voll ausgenutzt sind oder unter der Voraussetzung, daß die Bandeinheitenbelegung größer ist als ihre Verfügbarkeit. Das Planungsverfahren streicht im ersten Fall immer den (die) Job(s) mit der kleinsten Bandeinheitenanforderung und bei gleicher Anforderung den Job mit der kleineren Restzeit, der die Einhaltung der Kapazitätsgrenzen gewährleistet.

Die Auswahl e i n e s zu streichenden Jobs im zweiten Fall zeigt Abb. 40, die im folgenden näher erläutert wird.

Abb. 40 : Ablaufdiagramm zur Auswahl e i n e s zu streichenden Jobs unter der Voraussetzung, daß die Bandeinheitenbelegung größer ist als die Verfügbarkeit



Steht nur eine geringere Anzahl von Bandeinheiten zur Verfügung als vorläufig durch das Planungsverfahren belegt wurden, wird versucht, einen Job x' zu streichen, der erstens die volle Ausnutzung der Bandeinheiten durch die restlichen Jobs und zweitens die Einhaltung der übrigen Kapazitätsgrenzen sichert. Wenn ein solcher Job nicht vorgeplant ist, wird ein Job x^* gestrichen, der die möglichst weitgehende Belegung der Bandeinheiten unter Beachtung der Kapazitätsverfügbarkeit aller übrigen Elemente gewährleistet. Vor der Jobauswahl müssen zunächst die Differenzen D zwischen der aus der Vorplanung resultierenden

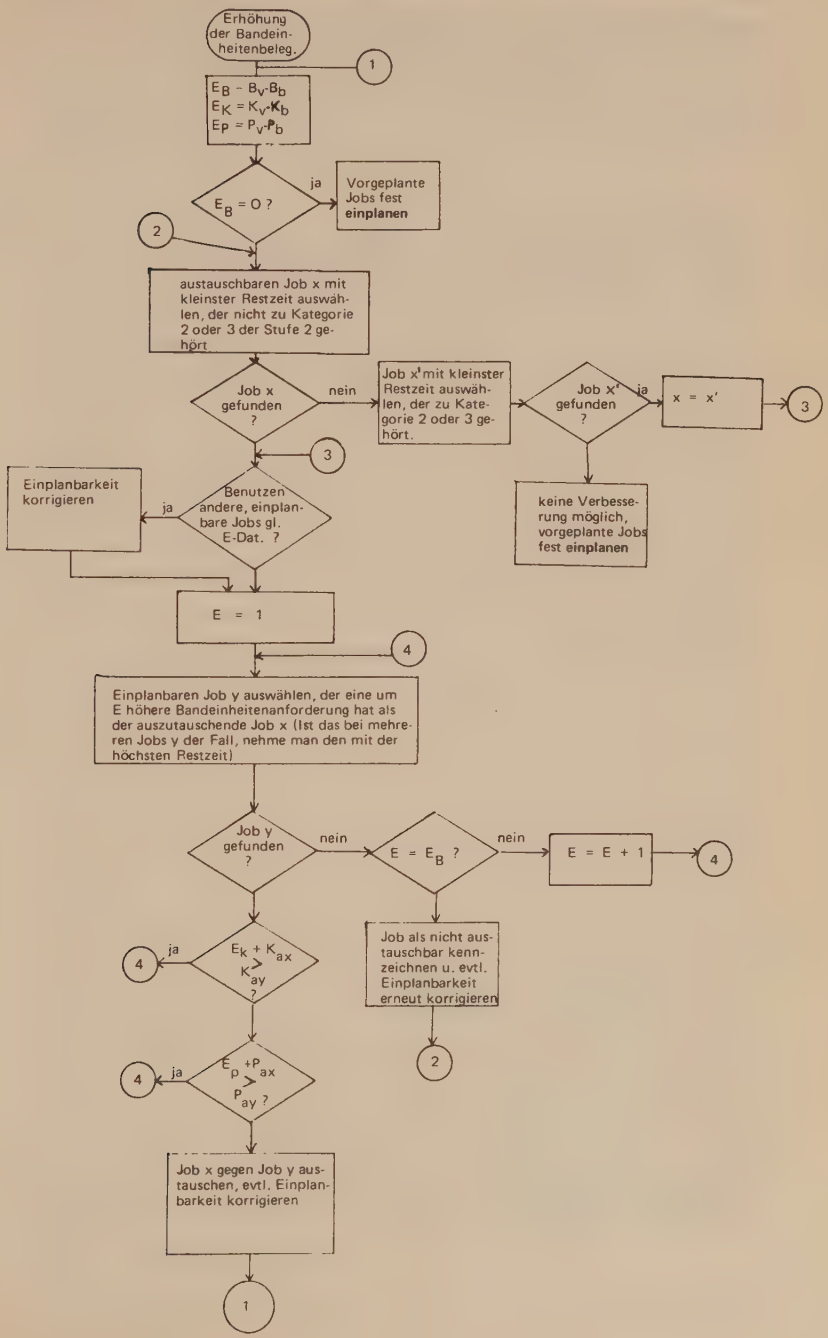
Belegung b und der Verfügbarkeit v der Bandeinheiten B ($D_B = B_b - B_v$), der Kernspeicher-Regions K ($D_K = K_b - K_v$) und der Plattenspeicher P ($D_P = P_b - P_v$) ermittelt werden. Für D_B muß sich in diesem Fall ein positiver Wert ergeben, D_K und D_p können negativ, gleich Null oder positiv sein. Dann werden die vorgeplanten Jobs x ($x = 1, \ldots, j_t$) durchlaufen, und es wird bei jedem Job geprüft, ob seine Bandeinheitenanforderung B_{ax} größer oder gleich der Differenz D_B ist. Bei Gleichheit wird geprüft, ob durch die Streichung dieses Jobs die Kapazitätsgrenzen der Plateneinheiten ($P_{ax} \geq D_p$) und der Kernspeicher-Regions ($K_{ax} \geq D_k$) gewahrt bleiben, und wenn dies geschieht, wird die Restzeit des Jobs x mit der Restzeit eines eventuell bereits gefundenen Jobs x' verglichen. Der Job mit der kleineren Restzeit wird zur Streichung vorgemerkt. Ist $B_{ax} > D_B$, wird dieser Job nur untersucht, wenn noch kein Job gefunden wurde, bei dem $B_{ax} = D_B$ ist. Zur Streichung vorgemerkt wird hier allerdings nicht der Job mit der kleineren Restzeit, sondern der Job, der eine bessere Ausnutzung der Bandeinheiten verspricht. Ist eine der genannten Bedingungen nicht erfüllt oder hat man einen Job zur Streichung vorgesehen, wird der nächste Job des Vorplans untersucht usw., bis alle Jobs durchlaufen sind ($x = j_t$). Das Planungsverfahren streicht dann entweder den unter x' vorgemerkten Job (Bedingung: $x' \neq 0$) und plant die restlichen Jobs sofort fest ein, weil die Bandeinheiten vollständig belegt sind oder streicht den unter x^* vorgesehenen Job (Bedingung: $x^* \neq 0$) und versucht anschließend die Bandeinheitenbelegung zu erhöhen. Konnte kein einzelner Job gefunden werden, durch dessen Streichung die Einhaltung der Kapazitätsgrenzen garantiert wird, wählt das Verfahren zwei oder mehrere Jobs nach dem gleichen Prinzip zur Streichung aus. Nach der so vorgenommenen Korrektur des Vorplans muß vom Planungsverfahren die Liste der einplanbaren Jobs eventuell um die Jobs erweitert werden, die die gleichen Eingabedateien benötigten wie der gestrichene Job und die nach seiner vorläufigen Einplanung zunächst als nicht mehr einplanbar gekennzeichnet worden waren.

Der Teil des Planungsverfahrens, der in der dritten Stufe eine Steigerung der Bandeinheitenbelegung anstrebt, braucht selbstverständlich nur durchlaufen zu werden, wenn noch einplanbare Jobs verfügbar sind. Zusätzliche Jobs können bei der Erhöhung nicht mehr eingeplant werden; es erfolgt nur ein Austausch zwischen den vorgeplanten Jobs und den Jobs, die eine höhere Anforderung an Bandeinheiten aufweisen. Abb. 41 zeigt das Ablaufdiagramm zur Verbesserung der Bandeinheitenbelegung. Durch die Streichung eines Jobs oder auf Grund des Vorplanungsergebnisses ist sichergestellt, daß die Kapazitätsverfügbarkeit bei keinem Systemelement überschritten ist. Die Differenz zwischen verfügbarer und belegter Kapazität gibt dann den jeweiligen Erhöhungsspielraum für die Magnetbandeinheiten E_B ($E_B = B_v - B_b$), für die Kernspeicher-Regions E_K ($E_K = K_v - K_b$) und für die Platteneinheiten E_P ($E_P = P_v - P_b$) an. Aus den vorgeplanten Jobs wird der Job x ($x = 1, \ldots, i_t$) mit der kleinsten Restzeit ausgewählt, der erstens nicht zur Kategorie zwei oder drei der zweiten Stufe des Verfahrens[1] gehört und der zweitens noch nicht das Erhöhungsverfahren durchlaufen hat. Findet sich ein solcher Job nicht, wird ein nicht kritischer Job mit Abgabetermin (Stufe 3) oder ein subkritischer Job (Stufe 2) ausgewählt. Da Job x gegen einen Job mit höherer Bandanforderung ausgetauscht werden soll, können nach der Vorplanung des Jobs x zunächst gesperrte Jobs, die gleiche Eingabedateien wie Job x anfordern, bei der Erhöhung zusätzlich berücksichtigt werden. Der Erhöhungsparameter E ($E = 1, \ldots, E_B$) gibt an, in welcher Höhe die Bandeinheitenanforderung des neu einzuplanenden Jobs y über der alten Bandeinheitenanforderung des Jobs x liegen kann. Aus den Jobs, die eine um E höhere Anforderung haben als Job x und die die Kapazitätsverfügbarkeit des Kernspeichers ($K_{ay} < E_k + K_{ax}$) sowie der Plattenspeichereinheiten ($P_{ay} < E_p + P_{ax}$) nicht überschreiten, wird der Job y mit der höchsten Restzeit gegen den vorgeplanten Job x ausgetauscht. Findet sich kein Job mit einer höheren Bandeinheitenanforderung, kennzeichnet das Planungsverfahren ihn als nicht austauschbar und sperrt evtl. erneut die einplanbaren Jobs, die die gleichen Eingabedateien

---

1) vgl. oben S. 177.

Abb. 41 : Ablaufdiagramm zur Verbesserung der Bandeinheitenbelegung



benutzen. Wird der Austausch dagegen vorgenommen und ist E_B noch ungleich Null, sucht das Verfahren wieder den Job mit der kleinsten Restzeit aus und versucht ihn durch einen Job mit höherer Anforderung zu ersetzen usw.

Dieser Prozeß wird solange fortgesetzt, bis entweder alle Jobs als nicht austauschbar gekennzeichnet sind oder die verfügbare Anzahl von Bandeinheiten erreicht ist. Durch dieses Vorgehen ist sichergestellt, daß für einen vorgegebenen Zeitpunkt stets die Jobs fest eingeplant werden, die die größtmögliche Belastung des Engpaßelements gewährleisten und die die höchste Restzeit besitzen.

Im Anschluß an die Einplanung werden vom Planungsverfahren die Jobabhängigkeiten berichtigt, d.h. es wird überprüft, ob durch die Einplanung Folgejobs einplanbar geworden sind und zu welchem Zeitpunkt sie frühestmöglich beginnen können. Außerdem wird für die nächsten Zeitpunkte des Zeitgebers die Verfügbarkeit der einzelnen Systemelemente festgestellt. Durch den Vergleich mit den Kapazitätsanforderungen der einplanbaren Jobs erreicht man, daß das Planungsverfahren erst dann wieder einzuplanen versucht, wenn die minimal benötigten Kapazitäten verfügbar sind. Die Einplanung wird solange fortgesetzt, bis entweder alle Jobs eingeplant sind oder die obere Grenze des Belegungsplanungszeitraums erreicht ist.

In der gleichen Form wie es bisher für das Engpaßelement Magnetbandeinheiten dargestellt wurde, arbeitet das Planungsverfahren auch, wenn ein anderes Systemelement als Engpaßfaktor zu betrachten ist. Es plant vorläufig ein, verwirft bei Vorliegen bestimmter Bedingungen[1] diesen Plan wieder, versucht ihn zu verbessern und plant endgültig erst ein, wenn für den betrachteten Zeitpunkt mit vertretbarem Aufwand keine günstigere Jobkombination gefunden wird. Das Optimierungziel, nämlich die Minimierung der Gesamtverarbeitungszeit wird dabei erstens durch die möglichst vollständige

---

1) vgl. oben S. 186.

Belegung des Engpaßelements, das nach einer Neuberechnung der Belegungsfaktoren auch während des Einplanungsprozesses wechseln kann, zweitens durch die durchsatzorientiert erfolgende Auswahl der Jobs für den Vorplan und drittens durch die Auswahl der Jobs nach dem Kriterium der maximalen Restzeit verfolgt.

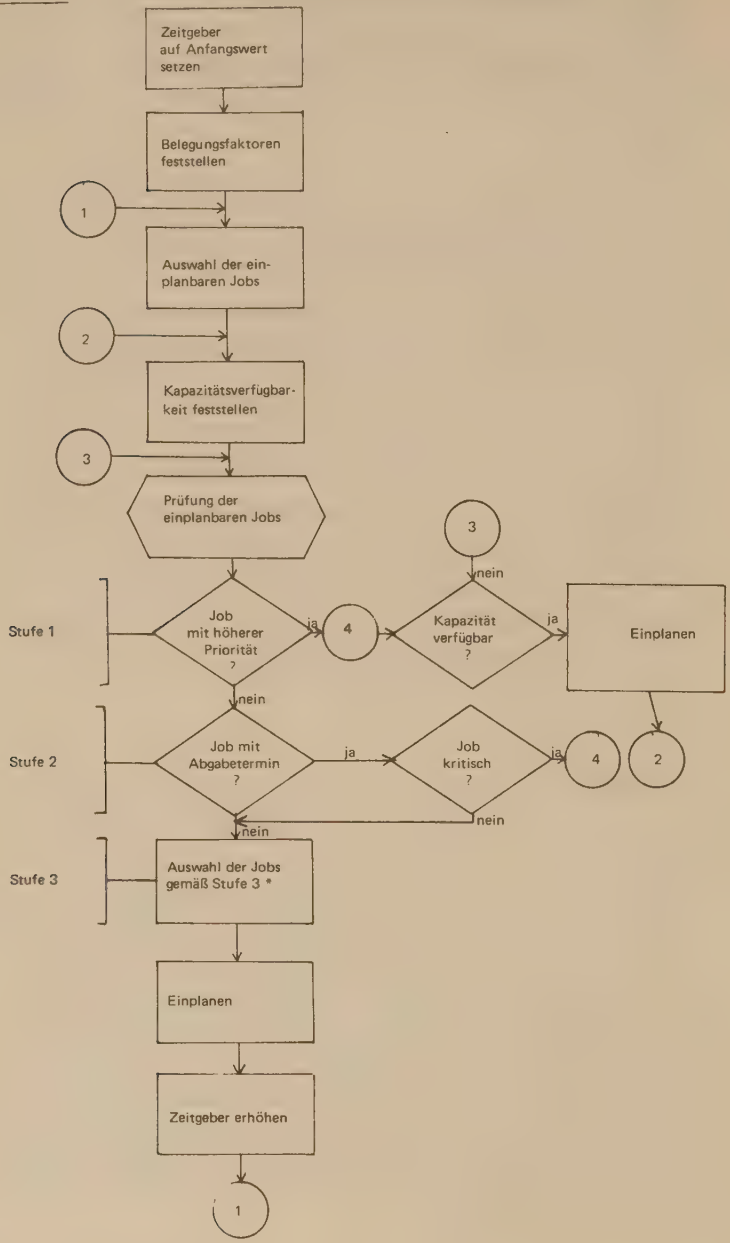
6.4 Die Effizienz des Planungsverfahrens

Die Leistungsfähigkeit von heuristischen Verfahren ist in der Regel nur schwer zu beurteilen, da sie häufig im Hinblick auf ganz bestimmte Problemstellungen entwickelt wurden und ein Vergleich mit der tatsächlichen Optimallösung nur in den seltensten Fällen möglich ist. Als allgemeingültige Kriterien für ihre Beurteilung kommen 1. der Umfang der erforderlichen Dateneingabe, 2. die Schnelligkeit, mit der das Verfahren arbeitet, und - soweit feststellbar - 3. die Güte des Lösungsvorschlags in Betracht. An Hand dieser Punkte soll daher auch das vom Verfasser entwickelte Planungsverfahren auf seine Wirksamkeit überprüft werden, soweit dies auf Grund der bisherigen Erfahrungen schon möglich ist.

Zu 1:

Zur Zeit können bis zu 200 Jobs und drei Systemelemente mit dem zu diesem Verfahren vom Verfasser geschriebenen Programm eingeplant werden. Die maximale Dauer des Belegungsplans beträgt 24 Stunden. Abb. 42 zeigt ein stark vereinfachtes, integriertes Ablaufdiagramm des Planungsprogramms. Wie beschrieben ist die sofortige Einplanung von Jobs mit einer höheren Priorität möglich (Stufe 1), vorgegebene Abgabe- und Bereitstellungstermine werden beachtet (Stufe 2). Die Prioritätsregeln, nach denen es einplant, versuchen eine möglichst vollständige Belegung des Engpaßelements bei gleichzeitig möglichst weitgehender Ausnutzung der verfügbaren Regions durchzuführen. Bei gleicher Anforderung eines Systemelements durch einen Job wird jeweils der Job mit der höheren Restzeit eingeplant (Stufe 3).

Abb. 42: Integriertes Ablaufdiagramm des Planungsprogramms



* vgl. S. 178 - 193

Die für die Belegungsplanung benötigten Daten werden bei der Tagesplanung[1] aufbereitet, die die einzelnen Jobinformationen wiederum aus der Jobstammdatei entnimmt, so daß sich die manuelle Eingabe für das Planungsverfahren auf die Angabe der verfügbaren Kapazität beschränken kann.

Wird die Belegungsplanung zum wiederholten Male während eines Planungszeitraums durchgeführt, müssen zur Zeit zusätzlich noch Löschkarten für die bereits erledigten Jobs eingegeben werden. In einer weiteren Ausbaustufe ist geplant, daß das Belegungsplanungs-Programm die erledigten Jobs den Logbuch-Routinen des Betriebssystems automatisch entnimmt.

Zu 2:
In etwa 20 Testläufen[2] mit dem in Abb. 33 aufgeführten Beispiel stellte sich heraus, daß das Verfahren außerordentlich schnell arbeitet. Die 35 Jobs wurden durchschnittlich in einer CPU-Zeit von 4 Sekunden und einer Verweilzeit von 45 Sekunden eingeplant. Beim Einsatz des Verfahrens für die tägliche Belegungsplanung eines Rechenzentrums ist die Anzahl der einzuplanenden Jobs zwar wesentlich höher als in diesem Beispiel, der Belegungszeitraum kann aber auf max. 5 Stunden begrenzt werden, da der Belegungsplan erfahrungsgemäß nach etwa 3 Stunden neu erstellt werden muß, weil die Verweilzeitschätzungen der einzelnen Jobs in zu großem Umfang von den tatsächlich auftretenden Zeiten abweichen. Für einen Zeitraum von 5 Stunden müssen dann nur etwa 40 - 50 Jobs eingeplant werden, und eine wesentliche Erhöhung der genannten Zeiten ist bei dieser Größenordnung nicht zu erwarten.

Zu 3:
Es bietet sich an, die Güte eines Lösungsvorschlages an den Anschaltzeiten zu messen, die bei einer Steuerung der EDV-Anlage

---

1) vgl. oben S. 71 - 104.

2) Dabei wurden die Kapazitätsanforderungen und die Verweilzeiten der Jobs variiert sowie Bereitstellungs- und Abgabetermine vorgegeben.

durch den Konsoloperator voraussichtlich auftreten würden. In Tab. 28 ist die Einplanungsunterlage eines Operators für das Beispiel aus Abb. 33 dargestellt, die auf Grund der Angaben der Sachbearbeitungs- bzw. der Arbeitsvorbereitungsabteilung erstellt wird.

Tab. 28 : Einplanungsunterlage eines Konsoloperators in einem Rechenzentrum ohne Belegungsplanung für das Beispiel aus Abb. 33

Block	Jobname	Text	Verweilzeit	Druckzeit	Band-einheiten
1	1		10		2
	2		15		3
	3		10		1
	4		30		3
	5		20		1
2	6		30		2
	7		40		3
	8		60		4
	9		10		4
	10		20		1
	11		10		2
	12		20		3
	13		25		4
	14		15		3
	15		20		1
3	16		150		2
4	17		20		3
	18		20		4
	19		30		3
	20		10		4
	21		15		3
	22		20		2
	23		15		1
	24		30		2
	25		10		4
	26		20		2
5	27		50		3
	28		50		2
	29		60		1
	30		20		1
	31		100		2
	32		20		3
	33		30		4
	34		20		3
	35		30		2

Die logischen Jobabhängigkeiten ergeben sich bei der Planungsunterlage ausschließlich aus der Reihenfolge der Jobs; Parallellaufmöglichkeiten werden nur zwischen den Jobblöcken wahrgenommen. Die um die Jobs (14, 15), (18, 19, 20) und (28, 29) gezeichneten Klammern zeigen dem Operator an, daß die Erledigungsreihenfolge dieser Jobs beliebig von ihm festgelegt werden kann.

Abb. 43 zeigt den Belegungsplan für Magnetbandeinheiten, der sich wahrscheinlich nach der Ausführung der Jobs ergeben würde, wenn drei Jobs parallel zueinander erledigt werden können.

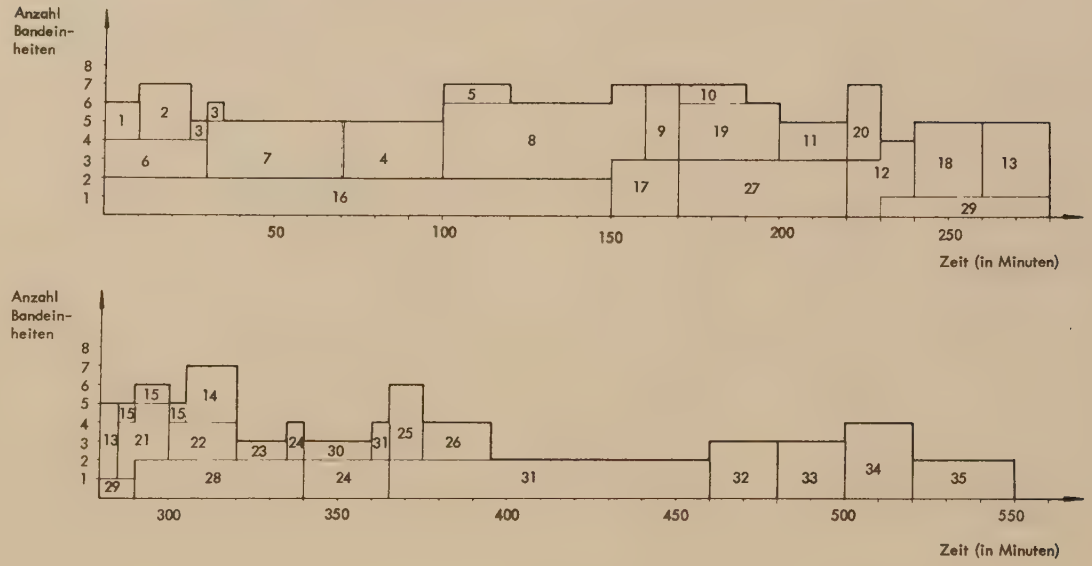
Der Operator würde danach von Block 1 den Job 1, von Block 2 den Job 6 und von Block 3 den Job 16 zuerst starten, nach Erledigung von Job 1 mit Job 2 fortfahren usw., wie es aus der Abbildung hervorgeht. Die 35 Jobs wären nach 550 Minuten erledigt. Der Ausnutzungsgrad der Kernspeicher-Regions liegt bei 60,3 %, der Ausnutzungsgrad der Bandeinheiten bei 65,6 % [1].

Abb. 44 zeigt demgegenüber den Belegungsplan, der mit Hilfe des Planungsverfahrens ermittelt wurde. Erfolgt die Abarbeitung der Jobs in der Reihenfolge, die das Planungsverfahren vorschlägt, können die 35 Jobs bereits nach 400 Minuten ausgeführt sein. Der Ausnutzungsgrad der Kernspeicher-Regions liegt hier bei 87,9 %, der der Bandeinheiten bei 91 %. Die Zeitersparnis hinsichtlich der Belegungsplandauer beläuft sich in diesem Beispiel auf 150 Minuten. Dieses Ergebnis darf sicherlich nicht überbewertet werden, es deutet aber zumindest an, daß das Planungsverfahren zu einer wesentlich besseren Ausnutzung der Systemelemente führt als dies bei manueller Steuerung möglich ist. Auch im Vergleich zu den Ergebnissen, die an Hand desselben Beispiels mit verschiedenen Prioritäts-

---

1) Zur Definition des Ausnutzungsgrades der Systemelemente vgl. oben S. 112.

Abb. 43 : Voraussichtliches manuelles Einplanungsergebnis für die Bandeinheiten auf Grund der Einplanungsunterlage in Tab. 28 durch einen Operator



regelkombinationen bei dem IBM-Programm PMS-IV erzielt wurden [1], erweist sich das Planungsverfahren als gut. Die kürzeste Belegungsdauer, die mit dem Programm PMS-IV erzielt wurde, lag bei 410 Zeiteinheiten [2].

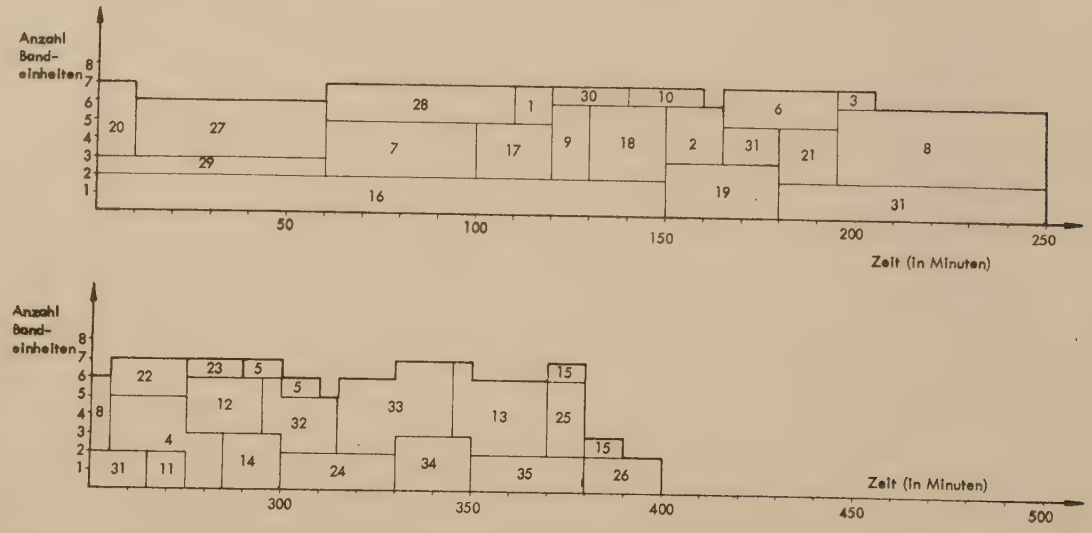
Obwohl das Beispiel in Abb. 44 eine für ein Rechenzentrum typische Jobstruktur aufweist und insofern erste Hinweise für die Effizienz des Planungsverfahrens geben kann, sollten endgültige Aussagen über das Verfahren erst gemacht werden, wenn es längere Zeit in der Praxis erprobt wurde.

---

1) vgl. oben S. 149.

2) Die CPU-Zeit von PMS-IV betrug dabei ca. 25 Sekunden, die Verweilzeit ca. 15 Minuten.

Abb. 44 : Belegungsplan für das Beispiel in Abb. 32 auf Grund des vom Verfasser entwickelten Planungsverfahrens , bei max. drei verfügbaren Regions und sieben Bandeinheiten



7 DISKUSSION EINIGER IM ZUSAMMENHANG MIT DER EINFÜHRUNG DES ABLAUF- UND BELEGUNGSPLANUNGSVERFAHRENS AUFTRETENDER PROBLEME

Bereits in der Einleitung wurde darauf hingewiesen, daß das Ziel dieser Arbeit nicht nur in der Entwicklung theoretisch fundierter Methoden liegen kann, sondern daß auch die Einsatzmöglichkeiten der vorgeschlagenen Planungsinstrumente und -verfahren im täglichen Ablauf eines Rechenzentrums als wesentlicher Punkt der Untersuchungen anzusehen ist. Eine kritische Betrachtung der Probleme, die bei der Einführung des Ablauf- und Belegungsplanungsverfahrens in der Praxis aufgetreten sind bzw. sich bereits absehen lassen, sollte zwar weitgehend Fragen behandeln, die in allen Rechenzentren in etwa in der gleichen Form auftreten, sie wird sich letztlich aber doch am organisatorischen Ablauf des Rechenzentrums ausrichten müssen, in dem diese Verfahren angewendet werden.

In diesem Kapitel werden Lösungsmöglichkeiten für fünf Problemkreise aufgezeigt, die teilweise abhängig voneinander sind:

1. Ermittlung des Anfangszustandes der EDV-Anlage bei einer wiederholten Durchführung der Belegungsplanung.
2. Auswirkungen eines Jobabbruchs auf die Belegungsplanung.
3. Berücksichtigung des Time-Sharing und der Datenfernverarbeitung bei der Belegungsplanung.
4. Ungenauigkeit der Jobplanung bei der Ausnutzung von Magnetbandeinheiten und Magnetplattenspeichern.
5. Festlegung von Jobverarbeitungszeiten (Verweilzeiten).

Bevor diese Probleme, die fast ausschließlich im Zusammenhang mit der Belegungsplanung auftreten, im einzelnen erörtert werden, sollen jedoch noch kurz einige Hinweise für die Durchführung der Jobablaufplanung (Tagesplanung) gegeben werden. Es hat sich herausgestellt, daß der Aufbau und die Pflege der Jobstammdatei ohne besondere Schwierigkeiten durchzuführen ist, wenn die Zielvorstellung, die mit ihr verfolgt werden soll, fixiert ist. Es muß allerdings darauf geachtet werden, daß zwischen der Aufnahme neu übergebener Routinejobs und ihrer ersten Erledigung eine Zeitspanne von etwa einem Arbeitstag zur Verfügung steht. In der Regel wird man neue Jobs bis 15 Uhr annehmen, um sie in einem mehrschichtigen Arbeitsbetrieb im Laufe der Nacht, in einem einschichtigen Betrieb zu Beginn des nächsten Tages in die Jobstammdatei einzuspielen.

Die Probleme der Ablaufplanung liegen hauptsächlich auf dem Gebiet der Terminfestlegung für die einzelnen Jobs. Hier ist wahrscheinlich ein sich über mehrere Monate erstreckender Lernprozeß notwendig, bis sich die Sachbearbeitungs- bzw. die Arbeitsvorbereitungsabteilung und die Fachabteilungen an eine größere Termintreue gewöhnt haben. Eine genaue Vorgabe und Einhaltung der Lauftermine wird sich am ehesten erreichen lassen, wenn man sowohl die Festlegung als auch die Überwachung der Einhaltung der Lauftermine einer besonderen Termingruppe überträgt[1]. Diese Gruppe sollte auch die notwendigen Unterlagen zur Auflösung von aufgetretenen Zyklen besitzen.

Es empfiehlt sich, zusätzlich zur Belegungsplanung, die in einem etwa 3-stündigen Rhythmus erfolgen muß, in jeder Planungsperiode auch zweimal täglich die Ablaufplanung[2] durchzuführen, da das Maschinenbedienungspersonal bei Ungültigkeit des Belegungsplans die Anlage dann an Hand dieser Unterlage steuern kann. Meist wird dieser Plan vor Einführung

---

1) Nach den Erfahrungen des Verfassers genügen für diese Gruppe zwei Arbeitsvorbereiter.

2) vgl. oben Abb. 34, S. 156.

der eigentlichen Belegungsplanung ohnehin in einer längeren Übergangsphase als Steuerungsinstrument verwendet werden, so daß der Konsoloperator sich in dieser Zeit bereits an das Lesen des Plans gewöhnen kann. Mit Hilfe einer Arbeitsanweisung sollte sichergestellt werden, daß die Erledigung der Arbeiten auch in diesem Zeitraum nach einfachen Prioritätsregeln geschieht, die eine günstige Ausnutzung der Anlage gewährleisten.

Für die Planung und Implementierung eines Belegungsplanungssystems rechnet der Verfasser mit der Zeit von etwa einem Mann-Jahr. Aber auch nach der Fertigstellung der entsprechenden Programme bedarf dieses System einer ständigen Kontrolle und Wartung, und auch der für ein Rechenzentrum günstigste organisatorische Ablauf im Hinblick auf die Durchführung der Belegungsplanung wird sich nur langsam herausbilden. Die Job-Control-Steuerkarten des Programms für das Planungsverfahren wird man voraussichtlich als "katalogisierte Prozedur" speichern,um den Abruf dieses Jobs ohne zeitliche Verzögerung direkt von der Konsole der Anlage vornehmen zu können.

Bei der wiederholten Durchführung der Belegungsplanung im Laufe eines Tages muß dem Programm für die Belegungsplanung mitgeteilt werden, von welchem kapazitiven Anfangszustand das Planungsverfahren ausgehen kann und welche Jobs bereits erledigt sind. Diese Jobs müssen in der Planungsdatei, die vom Ablaufplanungsprogramm erstellt wird[1)], gestrichen werden. Der kapazitive Anfangszustand ergibt sich aus den Jobs, die parallel zu der erneuten Durchführung der Belegungsplanung noch auf Grund des alten Belegungsplans ausgeführt werden. Aus den Jobnamen und den Startzeitpunkten der laufenden Jobs kann das Programm zur Belegungsplanung das voraussichtliche Ende dieser Jobs und die zu Beginn des neuen Belegungsplans verfügbare Kapazität der EDV-Anlage ermitteln und die Einplanung unter Berücksichtigung der sich durch die erledigten Jobs geänderten Abhängigkeiten vornehmen. Die Informationen über erle-

1) vgl. dazu S. 101.

digte und zur Zeit des Belegungsplanungslaufs im Ausführungsstadium stehende Jobs können dabei entweder über Lochkarten eingegeben oder direkt den Logbuch-Dateien des Betriebssystems entnommen werden.

Auf Grund von Hardware- oder Programmfehlern tritt ab und zu der Fall ein, daß ein Routinejob nicht zum normalen Ende kommt. Ein solcher Jobabbruch hat erhebliche Auswirkungen auf den Belegungsplan, es sei denn, er tritt erst am Ende eines Jobs auf, der keine Folgejobs freigibt. Zunächst muß auf jeden Fall an Hand der Ablaufplanungsunterlagen überprüft werden, welche Abhängigkeiten durch das nicht normale Ende des Jobs unterbrochen werden. Besitzt der Job Folgejobs, die direkt im Anschluß an seine Beendigung eingeplant sind, ist der Belegungsplan hinfällig, wenn nicht die entstehenden Lücken durch Programmumwandlungen (Compile- und Linkage-Editor-Läufe) oder passende Programmtests ausgeglichen werden können, die ja bei der Belegungsplanung zunächst unberücksichtigt geblieben sind, weil die Planungsdatei über diese Programme keinerlei Informationen besitzt. Durch das Einschieben dieser Arbeiten in den Belegungsplan wird zwar die optimale Zuordnung der Systemelemente zeitweilig unterbrochen, es erspart jedoch eine zu häufige Neudurchführung der Belegungsplanung. Da der Belegungsplan auf Grund falscher Vorausschätzungen der Verweilzeiten von Zeit zu Zeit ohnehin seine Gültigkeit verliert, bietet dieses Vorgehen eine Möglichkeit, seine Aussagefähigkeit bis zu einem solchen Zeitpunkt zu erhalten. Die Ausweichmaßnahmen sind wahrscheinlich nicht erforderlich, wenn der abgebrochene Job keinen Folgejob besitzt bzw. die Erledigung von Folgejobs erst am Ende eines Belegungsplans vorgesehen ist.

Da in der Zukunft das Time-Sharing und die Datenfernverarbeitung stark an Bedeutung gewinnen werden, steht jedes Rechenzentrum früher oder später vor der Aufgabe, diese Verarbeitungsformen bei der Belegungsplanung zu berücksichtigen. Das Time-Sharing-Verfahren[1] kann schon allein auf Grund der mit ihm verbunden Form der

---

1) vgl. oben S. 34.

Dialogverarbeitung nicht geplant werden und muß nur durch die kapazitiven Einschränkungen berücksichtigt werden, die sich durch seinen Einsatz für die Einplanung der Jobs der Stapelverarbeitung ergeben. Bei der Datenfernverarbeitung besteht dagegen die Möglichkeit, sie in die Belegungsplanung einzubeziehen, wobei allerdings vorausgesetzt wird, daß die Fernverarbeitung in ähnlicher Weise wie die Stapelverarbeitung durchgeführt wird (Remote-Batch-Processing). Ist es möglich, bereits am Vortag festzulegen, welche Jobs über Fernleitungen erledigt werden müssen, können sie bei der Ablauf- und Belegungsplanung unter Beachtung vorgegebener Prioritäten, Bereitstellungs- oder Abgabetermine berücksichtigt werden. Daraus ergäbe sich der Vorteil, daß die von diesen Jobs in Anspruch genommenen Kapazitäten bei der Einplanung der normalen Jobs nicht subtrahiert zu werden brauchen. Diese Möglichkeit besteht dagegen nicht, wenn die Jobs der Datenfernverarbeitung Echtzeitcharakter besitzen, d.h. sofort nach ihrem Eintreffen im System mit höchstem Vorrang erledigt werden müssen.

Die Einplanung von Jobs und nicht von Jobsteps bedingt eine gewisse Ungenauigkeit des Belegungsplans. In der Jobstammdatei werden für jeden Job die insgesamt höchsten Kapazitäten der einzelnen Steps gespeichert. Benötigt z.B. ein Step 1 vier Magnetbandeinheiten und 10 Cylinder Magnetplattenspeicherplatz, Step 2 jedoch nur eine Bandeinheit aber 100 Cylinder Speicherplatz, belegt der entsprechende Job in der Belegungsplanung vier Bandeinheiten und 100 Cylinder. Läuft Step 2 zusätzlich sehr lange, wird ersichtlich, daß drei Magnetbandeinheiten für die Dauer dieses Steps praktisch verschenkt werden. Bereits in Kapitel 5.1 wurde aber darauf hingewiesen, daß man diesen Nachteil in Kauf nehmen sollte, um die dem Ablauf im Rechenzentrum entsprechende Jobplanung nutzen zu können. Zur Erzielung einer gleichmäßigen Nutzung dieser Ungleichmäßigkeiten bietet sich an, zunächst durch eine Analyse der einzelnen Jobs festzustellen, in welchen Programmen eine besonders krasse Diskrepanz auftritt, um dann diese Jobs eventuell in mehrere kleinere Jobs mit einer ausgeglicheneren Anforderung zu zerlegen.

Das zweifellos größte Problem, das im Zusammenhang mit der Einführung der Belegungsplanung in der in Kapitel 6 vorgeschlagenen Form auftritt, besteht in der realistischen Festlegung der Verarbeitungszeiten (Verweilzeiten) von Jobs, die im Multiprogrammbetrieb erledigt werden. Durch die verzahnte Abarbeitung der einzelnen Programme in der Zentraleinheit kann die Verarbeitungszeit eines Jobs bei jedem Lauf mehr oder weniger stark schwanken und damit die Aussagefähigkeit des Belegungsplans in Frage stellen. Dieses Problem wird sich zwar in Multiprogramming-Systemen nicht vollständig lösen lassen, es gibt jedoch Möglichkeiten,die Schwankungsbreiten durch eine genaue Analyse der Verarbeitungszeiten eines Jobs über einen längeren Zeitraum hinweg einzuengen. Dazu ist es notwendig, die Verweilzeiten der Jobs näher zu untersuchen, die sich aus drei Einflußgrößen 1. CPU-Zeit, 2. Zeit für E-/A-Operationen und 3. WAIT-Zeit zusammensetzen. Der in kaufmännisch orientierten Rechenzentren bei weitem kleinste Teil der Verweilzeit entfällt auf die CPU-Zeit eines Jobs, wobei unter CPU-Zeit die Summe der Zeitspannen zu verstehen ist, in der ein Job während der Ausführung der einzelnen Tasks die Kontrolle über die Zentraleinheit besitzt. Die Zykluszeit der Zentraleinheit der IBM/370 - 165, d.h. die Zeitspanne, die die Zentraleinheit benötigt, um eine vorgegebene Anzahl von Bytes zu lesen und wieder zu regenerieren, beträgt beispielsweise nur 80 Nanosekunden und das bedeutet, daß ein Job im Durchschnitt schon nach Sekundenbruchteilen die Kontrolle über die Zentraleinheit wieder abgibt, um Lese- und/oder Schreiboperationen auszuführen. Die CPU-Zeit verändert sich hauptsächlich in Abhängigkeit von dem zu verarbeitenden Datenvolumen. Ebenso die zweite Einflußgröße, die die Höhe der Verweilzeit bestimmt. Sie setzt sich aus der Summe der Zeitintervalle zusammen, in der Ein-/Ausgabeoperationen vom Programm vorgenommen werden. Jeder READ- und jeder WRITE - Befehl bewirkt, daß eine Task die Kontrolle über die Zentraleinheit wieder verliert und unter Steuerung einer besonderen Routine des Betriebssystems Lese- oder Schreibvorgänge ausführt. Da die Datenübertragungsgeschwindigkeit der Ein-/Ausgabe-

kanäle wesentlich geringer ist und daher mehr Zeit beansprucht als die Zykluszeit des Kernspeichers, entfällt auf diese zweite Einflußgröße ein relativ großer Anteil an der Verweilzeit. Gleiches gilt für die dritte Einflußgröße, die sogenannte "WAIT" - Zeit, d.h. die Zeit, in der eine Task entweder auf die Zuordnung von Ein-/Ausgabekanälen oder auf die Zuteilung der Zentraleinheit wartet, denn in prioritätsgesteuerten Multiprogramming - Systemen werden die Jobs nach jeder Ein-/Ausgabeoperation erneut in eine - ihrer Priorität entsprechende - Warteschlange eingereiht, die vollständig durchlaufen werden muß, bevor ein Job wieder die Kontrolle über die Zentraleinheit erhält. Besonders wenn rechenintensive Jobs unter der gleichen Priorität wie E-/A-intensive Jobs mit in der Anlage ablaufen, kann diese WAIT-Zeit sehr hoch sein. Hier erweist sich das UNIVAC-Betriebssystem EXEC 8 allen anderen Systemen gegenüber als weit überlegen, da es die Ablaufprioritäten der einzelnen Jobs ständig verändert, um die WAIT-Zeiten klein zu halten.

Die Änderungen der Verarbeitungszeit eines Jobs von Lauf zu Lauf ist also von dem Umfang der zu verarbeitenden u n d der zu bewegenden Daten, der Ein-/Ausgabeintensität eines Jobs, der Jobpriorität und dem Job-Mix in der Anlage abhängig, d.h. von der E-/A-Intensität der Jobs, die zur gleichen Zeit erledigt werden.

Betrachtet man die Vielzahl dieser Einflußfaktoren und ihre möglichen Auswirkungen auf die Verarbeitungszeit eines Jobs, wird verständlich, daß diese Zeiten für ein und dasselbe Programm von Lauf zu Lauf stark variieren können. Nur eine genauere Analyse der einzelnen Jobs erlaubt deshalb eine Festlegung der Verweilzeit, die nicht zu sehr von der bei der Ausführung des Jobs auftretenden Verarbeitungszeit abweicht. Grundlage der Analyse muß eine sich über

mehrere Monate erstreckende Auswertung der Logbuch-Routinen sein, die eine erste Auskunft über die einzelnen Einflußfaktoren geben kann. Zusätzlich ist es notwendig, genaue Aufschreibungen über Art und Umfang der in einem Job benutzten Dateien und die Menge der in ihn hineinlaufenden Daten vorzunehmen. Diese Aufgabe sollte sowohl von der Sachbearbeitung als auch von der Termingruppe gelöst werden, die ständig in einem engen Kontakt zu den Fachabteilungen stehen.

Erste Auswertungen des Verfassers und der Termingruppe des Rechenzentrums, in dem die Belegungsplanung Anfang 1973 eingeführt wird, ergaben, daß es bei etwa 50 % der Jobs ohne Schwierigkeiten möglich ist, die Verweilzeiten recht genau anzugeben. Bei diesen Jobs traten Differenzen von etwa 3 - 5 Minuten auf. Da manche Jobs eine niedrigere, andere eine höhere Zeit als vorgeplant besitzen, werden sich die Schwankungen der Laufzeiten dieser Jobs in etwa ausgleichen und die Aussagefähigkeit des Belegungsplans nicht beeinträchtigen. Bei anderen Jobs waren die Differenzen in den Verweilzeiten zwar wesentlich größer, die einzelnen Läufe ließen sich aber in bestimmte Gruppen einteilen, die nicht sehr stark differierten. Der Ablauf dieser Jobs wurde meist über Parameterkarten gesteuert, die in Abhängigkeit von der gewünschten Auswertung völlig verschiedene Programmzweige aufriefen und so zu den unterschiedlichen Verweilzeiten führten. Wieder andere, täglich ablaufende Jobs wiesen im Monatsverlauf einen deutlichen Anstieg der Verweilzeiten auf, der auf das wachsende Datenvolumen zurückzuführen war und der sich ebenfalls durch die entsprechende Änderung der Verweilzeiten in der Jobstammdatei berücksichtigen läßt. Besonders große Differenzen traten bei Jobs mit sehr vielen Steps und einer entsprechend hohen Laufzeit auf. Für die Belegungsplanung empfiehlt es sich, diese großen Jobs in mehrere kleine Jobs mit kürzeren Verweilzeiten zu unterteilen, die in der Regel besser abzuschätzen sind.

Da die Aufgabe einer möglichst genauen Festlegung der Verweilzeiten in Rechenzentren ohne eine exakte Belegungsplanung bisher noch nicht bestanden hat, weil für diese Festlegung keine Notwendigkeit bestand, darf die Größenordnung der auftretenden Schwankungen nicht verwundern. Selbst bei einem Multiprogramming-Betrieb mit durchschnittlich 5 parallellaufenden Jobs lassen sich die Verarbeitungszeiten nach Erfahrungen des Verfassers für die Belegungsplanung aber hinreichend genau festlegen. Wie bei unvorhersehbaren Jobabbrüchen ist es auch hier bei kleineren Abweichungen vom Plan angebracht, entstehende Lücken durch Programmumwandlungsläufe (Compile- und Linkläufe) zu füllen. Bei Jobs, deren Verweilzeiten bekanntermaßen sehr stark differieren, können außerdem die frühest möglichen Starttermine ihrer Folgejobs um zwei Zeitgeberschritte, d.h. bei dem hier vorgeschlagenen Verfahren um zehn Minuten, verschoben werden, um eine den Abhängigkeiten entsprechende Erledigung der Jobs in der Ausführungsphase sicherzustellen. Erst wenn die Abweichungen vom Plan zu groß werden, ist eine Neuplanung notwendig. In der Regel dürfte dies etwa nach drei Stunden der Fall sein. In einem drei-schichtig betriebenen Rechenzentrum muß das Belegungsplanungsprogramm also etwa achtmal täglich laufen (Laufzeit insgesamt ca. 20 Minuten). Hinzu kommt das Jobablaufplanungsprogramm für die Erstellung des Tagevorplans und des eigentlichen Tagesplans (Laufzeit insgesamt ca. 40 Minuten). Einem Aufwand von etwa einer Stunde in einer Kernspeicher-Region - bezogen auf die reine Rechenzeit - steht damit bei Einführung des Belegungsplanungsverfahrens selbst bei einer vorsichtigen Schätzung eine Ersparnis der Anschaltzeit der Anlage von mindestens 10 - 15 % gegenüber.

8 SCHLUSSBETRACHTUNG

Die Einführung der Multiprogrammverarbeitung erlaubte zwar die wesentlich bessere Nutzung der hohen internen Geschwindigkeit der Zentraleinheiten, führte aber zu erheblichen Schwierigkeiten bei der Kombination der Jobs, die parallel zueinander ausgeführt werden sollten. Die Hersteller von Datenverarbeitungsanlagen begnügen sich bisher mit dem Hinweis, daß nur voneinander unabhängige Jobs gleichzeitig erledigt werden dürfen, ohne entsprechende Software-Programme für die - im Hinblick auf eine minimale Gesamtverarbeitungszeit - optimale Jobkombination zur Verfügung zu stellen. Aus dieser Problematik ergab sich das Anliegen der vorliegenden Arbeit, nämlich ein Verfahren für die automatisierte Belegungsplanung einer im Multiprogramming-Betrieb arbeitenden EDV-Anlage zu entwickeln . Es mußte neben den vielfältigen Restriktionen, die auf Grund von Datenabhängigkeiten, Bereitstellungs- und Abgabeterminen von Jobs auftraten, auch die Einplanungs-Routinen der Betriebssysteme berücksichtigen bzw. durfte ihnen zumindest nicht zuwiderlaufen. Über die Entwicklung ähnlicher Verfahren ist dem Verfasser weder in der Praxis noch in der Literatur etwas bekannt geworden[1].

Die Belegungsplanung konnte nur für solche Arbeiten vorgenommen werden, deren Lauftermine festlegbar waren, deren Umfang fest umrissen und deren Verarbeitungszeit somit abschätzbar war, sowie für Arbeiten, deren kapazitive Anforderungen an das System bekannt waren. Diese Kriterien treffen

---

1) Ausgenommen hierbei ist lediglich das von SIEMENS entwickelte System "Dynamische Anlagenbelegung DAB 3" (vgl. oben S. 46 - 47), das den Anforderungen eines Großrechenzentrums aber nicht gerecht zu werden vermag.

auf Programme (Jobs) zu, die im Rahmen der Stapelverarbeitung erledigt werden[1]. Das Time-Sharing-Verfahren sowie die Real-Time-Verarbeitung entziehen sich einer Planung durch den Anwender, weil erstens nicht bekannt ist, wann Programme abgerufen werden und zweitens die Verweilzeit der Programme in der Regel starken Schwankungen unterworfen ist. Die Datenfernverarbeitung kann berücksichtigt werden, soweit es sich um das Verfahren des "remote batch processing" handelt, die Jobs also ebenfalls wie normale Stapelverarbeitungsprogramme erledigt werden.

Aus Gründen des Datenschutzes sind die Hersteller von EDV-Anlagen bestrebt, EDV - Automaten zu entwickeln, die sich selbst steuern und die sich selbst verwalten. Durch die damit verbundene Anonymität des Ablaufs, die durch die größere Automatisierung hervorgerufen wird, deutet sich eine Möglichkeit des Datenschutzes an. In dieser Hinsicht war auch der Begriff der "automatisierten" Belegungsplanung zu verstehen, einer Planung also, die möglichst ohne Eingriffe von außen selbsttätig vom Rechner erstellt wird. Dieser Forderung wurde in der Arbeit durch den Aufbau eines integrierten Planungssystems Rechnung getragen. Grundlage dieses Systems ist die Jobstammdatei, in der alle zur Belegungsplanung benötigten Informationen über die einzelnen Jobs enthalten sind. Durch eine Auswertung der Terminsteuerworte werden die Jobs ermittelt, die am Planungstag laufen müssen, und durch den Vergleich der Ein-/Ausgabedatenbestände bzw. der im Feld "Folgejobs" gemachten Angaben stellt die EDV-Anlage selbsttätig die bei der Erledigung der Jobs einzuhaltende Reihenfolge fest (Jobablaufplanung). Die Jobabhängigkeiten, die Kapazitätsanforderungen der einzelnen Jobs sowie die Kapazitätsverfügbarkeit der Systemelemente werden als Eingabe für das Belegungsplanungsprogramm benötigt, das die eigentliche Kombination der Jobs unter Beachtung der angegebenen Restriktionen mit Hilfe des heuristischen Planungsverfahrens vornimmt.

---

1) Programmwandlungsläufe (Compile- und Linkläufe) wurden nicht betrachtet, weil ihre Einplanung völlig unproblematisch ist, Programmtestläufe entziehen sich der Einplanung, weil über ihre Laufzeit keine Aussage getroffen werden kann.

In diesem System ist es neben der Angabe der jeweils für die Belegungsplanung verfügbaren Kapazität nur notwendig, auf der Stufe der Jobablaufplanung eventuell aufgetretene Zyklen aufzulösen, neue Jobs über Abrufkarten in die Planung aufzunehmen und erledigte Jobs zu löschen.

Zwangsläufig mußten sich die vom Verfasser geschriebenen Programme an den Möglichkeiten des Betriebssystems ausrichten, das in dem Rechenzentrum eingesetzt wird, für welches die Belegungsplanung durchzuführen war. Eine Übertragung auf andere Betriebssysteme dürfte an Hand der allgemein gehaltenen Ausführungen zu den einzelnen Verfahren jedoch keine Schwierigkeiten bereiten.

Besonders der auf der dritten Stufe des Belegungsplanungsverfahrens vorgeschlagene Weg, bei dem das Verfahren während des Einplanungsprozesses wiederholt selbsttätig den verstärkt zu berücksichtigenden Engpaßfaktor festlegt, auf Grund dieser Festlegung für jeden betrachteten Zeitpunkt eine Vorplanung vornimmt und diese mit Hilfe einer Entscheidungstabelle vor der endgültigen Einplanung zu verbessern versucht, wurde nach Kenntnis des Verfassers auch im Rahmen der Kapazitätsausgleichsplanung bisher nicht beschritten.

Über den Rahmen der täglichen und damit kurzfristigen Belegungsplanung hinaus kann das Planungsverfahren in begrenztem Umfang auch zur Bestimmung mittelfristig [1] optimaler Systemkonfigurationen herangezogen werden. Während bei einer Verarbeitung im Uniprogramming die Kapazität einer EDV-Anlage an den Maximalanforderungen des größten Jobs ausgerichtet werden mußte, gibt es bei der Multiprogrammverarbeitung in der Regel nur sehr grobe Anhaltspunkte für eine ausgeglichene und an den bestehenden Jobstrukturen ausgerichtete Konfigurierung eines Systems. Erweitert man die Jobstammdatei um die mittelfristig zusätzlich anfallenden Arbeiten, indem man sie als Planjobs mit ihren geschätzten Verweilzeiten, Lauftermine und Kapazitätsanforderungen in die Datei aufnimmt, so können mit

---

1) vgl. oben S. 112.

den bestehenden und den erwarteten Jobs fiktive Belegungspläne aufgestellt werden, deren Gesamtdauern Auskunft darüber geben, ob die in Zukunft anfallenden Arbeiten in der zur Verfügung stehenden Zeit erledigt werden können. Zeigt sich bei diesen Belegungsplänen, daß ein bestimmtes Systemelement (beispielsweise Bandeinheiten) ständig Engpaßfaktor bei der Einplanung war, kann die Verfügbarkeit dieses Elements erhöht und der Belegungsplan neu aufgestellt werden. Als Ergebnis dieses Vorgehens erhält man die mit Hilfe des Planungsverfahrens minimal erreichbare Belegungsdauer bei verschiedenen Systemkonfigurationen unter Beachtung der bestehenden und der erwarteten Jobstrukturen des Rechenzentrums und kann die Konfiguration auswählen, die dem mittelfristig erwarteten Arbeitsanfall am besten gerecht wird.

Der Anwendbarkeit des Planungsverfahrens sind dabei allerdings enge Grenzen gezogen, weil es von den Verweilzeiten der Jobs ausgeht. Es wird beispielsweise versagen, wenn der Auslastungsgrad der Zentraleinheit oder der Ein-/Ausgabekanäle so groß wird, daß bei einer Erhöhung der peripheren Systemkapazitäten oder der Kernspeichergröße eine bedeutende Verlangsamung der Jobs eintritt. Die Erhöhung der Verweilzeiten würde zur Ungültigkeit der fiktiven Belegungspläne führen. Bei kleineren Änderungen der Systemkonfiguration kann das Planungsverfahren jedoch wertvolle Entscheidungsunterlagen liefern.

LITERATURVERZEICHNIS

AUTORENTEAM, Netzplantechnik - ein Fortbildungskurs im Medienverbund Fernsehen - Lehrbuch - Seminare, VDI-Verlag GmbH, Düsseldorf 1971

BALAS, E., An Additive Algorithm for Solving Linear Programs with Zero-One Variables, Operations Research, 1965, S. 517 - 546

BUDIL, W., Netzplantechnik - ein neues System zur wirkungsvollen Projekt- und Terminplanung, Zeitschrift für Organisation, 1965, S. 41 - 51

DAKIN, R.J., A Tree-Search Algorithm for Mixed Integer Programming Problems, The Computer Journal, 1965, S. 250 - 255

DIEBOLD DEUTSCHLAND GMBH, Einsatz und Stand der Datenverarbeitung in der Bundesrepublik Deutschland, Frankfurt/M., 1968

DIEBOLD DEUTSCHLAND GMBH, Möglichkeiten und Grenzen der elektronischen Datenverarbeitung, Teil 1 - 3, Frankfurt/M., 1970

DOIG, A. G., und LAND, A. H., An Automatic Method of Solving Discrete Programming Problems, Econometrica, 1960, S. 497 - 520

FEHLER, D., Die Variationen-Enumeration, ein Näherungsverfahren zur Planung des optimalen Betriebsmitteleinsatzes bei der Terminierung von Projekten, Elektronische Datenverarbeitung, 1969, Heft 10, S. 479 - 483

GEWALD, K., KASPER, K., SCHELLE, H., Netzplantechnik, Band 2: Kapazitätsoptimierung, München - Wien, 1972

GEWALD, K., WILLE, H., WEBER, H. D., Netzplantechnik, Band 1, Zeitplanung, 2. Aufl., München 1967

GÖTZKE, H., Netzplantechnik, Leipzig 1969

GOMORY, R. E., An Algorithm for Integer Solutions to Linear Programs, Recent Advances in Mathematical Programming, R. GRAVES and P. WOLFE, Editors, New York, 1963

GOMORY, R. E., All-Integer Programming Algorithm, IBM Research Center, Research Report RC - 189, 1960

IBM, Das IBM System /370 Modell 165, IBM Deutschland, Form-Nr. 15 822

IBM, System /360 Operating System, Concepts and Facilities, File No. S 360-20, Form C 28-6535-1, 2. Aufl., 1967

KASPER, K., GEWALD, K., SCHELLE, H., Netzplantechnik, Band 2: Kapazitätsoptimierung, München - Wien, 1972

KÖNIG, D., Theorie der endlichen und unendlichen Graphen, New York, 1950

KÜMMERLE, K., Charakteristische Größen zur Beschreibung der Leistungsfähigkeit und Effektivität von EDV-Anlagen, Elektronische Rechenanlagen, 1972, Heft 1, S. 12 - 18

LAND, A. H. und DOIG, A. G., An Automatic Method of Solving Discrete Programming Problems, Econometrica, 1960, S. 497 - 520

MÜLLER-MERBACH, H., Ein Verfahren zur Planung des optimalen Betriebsmitteleinsatzes bei der Terminierung von Großprojekten, AWF-Mitteilungen, 1967, Heft 2, S. 5 - 10

MÜLLER-MERBACH, H., Optimale Reihenfolgen, Berlin 1970

MÜLLER-MERBACH, H., Operations Research, Berlin 1969

SCHELLE, H. D., GEWALD, K., KASPER, K., Netzplantechnik Band 2: Kapazitätsoptimierung, München - Wien, 1972

SIEMENS, PBS 4004 - Dynamische Anlagenbelegung DAB, Best.-Nr. D 14/4144, München 1971

SIEMENS, PBS, Organisationsprogramm, München 1970

SIEMENS, System 4004, 4004/35, 135, 45, 55, 60, 150, Platte-Betriebssystem, Beschreibung, Best.-Nr. D 14/4014

STAHLKNECHT, P., Operations Research, 2. Aufl., Braunschweig 1970

UNIVAC, 1108 Multiprocessor System, Operating System EXEC 8, Programmers Reference, UP - 4144 Rev. 1, 1968

VOIGT, J. P., Termin- und Kapazitätsplanung für Instandsetzungs- und Montageprojekte durch Netzplantechnik, Stahl und Eisen, 1971, Heft 20, S. 1121 - 1129

WEBER, H.D., WILLE, H., GEWALD, K., Netzplantechnik, Band 1, Zeitplanung, 2. Aufl., München 1967

WEBER, K., Projektanalyse mit CPM, Industrielle Organisation, 1968, Heft 37, S. 236 - 248

WIEST, J. D., The Scheduling of Large Projects with Limited Resources, Ph. D. Dissertation, Carnegy Institute of Technology, Pittsburgh 1963

WILLE, H., GEWALD, K., WEBER, H. D., Netzplantechnik, Band 1, Zeitplanung, 2. Aufl., München 1967

WIRTSCHAFTSWOCHE, Nr. 29, Jahrgang 1971, S. 55 - 58

WIRTSCHAFTSWOCHE, Nr. 51, Jahrgang 1971, S. 25 - 27

WÖHE, G., Einführung in die Allgemeine Betriebswirtschaftslehre, 6. Aufl., Berlin 1964

WOLTERS, M. F., Der Schlüssel zum Computer, Textbuch 1, Düsseldorf 1969

ANHANG

Anhang 1 : Beispiel eines Ausdrucks der Monatsplanungsübersicht für einen bestimmten Tag

MONATSPLANUNG FUER DIE IBM /360-65 VOM 1. 6. 72

PLANUNGSTAG 7. 6.

MERK	ANZ. JOBS	VERARBEITUNGSKLASSEN										MERKSSCHLUESSEL									ANZ. LF	ZEITEN			
		1	B	C	D	E	F	G	H		1	2	3	4	5	6	7	8	9		VZ	GZ	DZ
2	52	0	27	19	6	0	0	0	0		4	2	18	1	0	27	0	0	0	22	2705	60	1485
3	4	0	4	0	0	0	0	0	0		0	1	3	0	0	0	0	0	0	0	80	0	0
4	2	0	1	1	0	0	0	0	0		2	0	0	0	0	0	0	0	0	2	235	15	20
5	6	0	3	3	0	0	0	0	0		0	3	0	0	0	0	0	0	3	2	160	0	40
12	2	0	2	0	0	0	0	0	0		1	0	0	0	0	1	0	0	0	2	85	0	0
13	2	0	1	1	0	0	0	0	0		0	1	1	0	0	0	0	0	0	0	95	0	120
20	1	0	1	0	0	0	0	0	0		0	0	1	0	0	0	0	0	0	0	20	0	0
24	4	0	2	2	0	0	0	0	0		0	0	4	0	0	0	0	0	0	2	65	5	10
30	2	0	0	1	1	0	0	0	0		0	2	0	0	0	0	0	0	0	1	40	0	0
32	3	0	1	1	1	0	0	0	0		0	3	0	0	0	0	0	0	0	1	170	140	190
36	9	0	3	4	2	0	0	0	0		0	0	9	0	0	0	0	0	0	2	190	0	105
906	1	0	0	1	0	0	0	0	0		0	0	1	0	0	0	0	0	0	0	10	0	0
907	3	0	1	2	0	0	0	0	0		0	0	3	0	0	0	0	0	0	1	510	0	220
909	2	0	1	1	0	0	0	0	0		0	0	2	0	0	0	0	0	0	1	95	0	30
923	1	0	0	1	0	0	0	0	0		0	0	1	0	0	0	0	0	0	0	5	0	0
924	6	0	0	6	0	0	0	0	0		1	0	0	0	0	0	0	0	0	2	320	40	45
SUMME	100	0	47	43	10	0	0	0	0		8	17	43	1	0	28	0	0	3				

SUMME DER VERARBEITUNGSZEITEN 38 4785 260 2265

Anhang 2 : Beispiel einer Listenausgabe für die Job-Ablaufplanung

TAGESPLANUNG VOM 4.9.1972 FUER DIE /370-165

(1) BLOCK 19													
(2) K	(3) N	(4) JOBNAME	(5) WERK	(6) PRG	(7) SA	(8) BEZEICHNUNG	(9) LF CL (15)	(10) D	(11) D-CL DR-A (16)	(12) MIN MIN (17)	(13) REG CYL (18)	(14) BER-T ABG-T (19)
U	3	J600483	2	218	13	HHWP PL AUFTRAGSEINGANG HALBZEUG PHOENIX	30	K	C	100		0
						FOLGEJOBS : (20) J600491 J600511 J600525	B		I	10	16	0
L	U	J600491	2	216	13	HHWP STW BB-HZ TAEGL. LISTEN TEIL 2	60	T	C	100		0
						GLEICHER E-FILE IN : (21) J600511 J600525	B		1	C	70	0
L	U	J600511	2	216	13	HHWP STW BB-H TAEGL. VERSANDAVIS	10	BL	C	100		0
						GLEICHER E-FILE IN : J600491 J600525	B		0	C	23	0
L	U	J600525	2	216	13	HHWP STW BB-HZ TAEGL. ABRD. U. BEST-E.	200	I	C	100		0
						GLEICHER E-FILE IN : J600491 J600511	B		1	C	66	0
						VERARBEITUNGS- UND DRUCKZEITEN DES BLOCKS	300			20		

Legende:

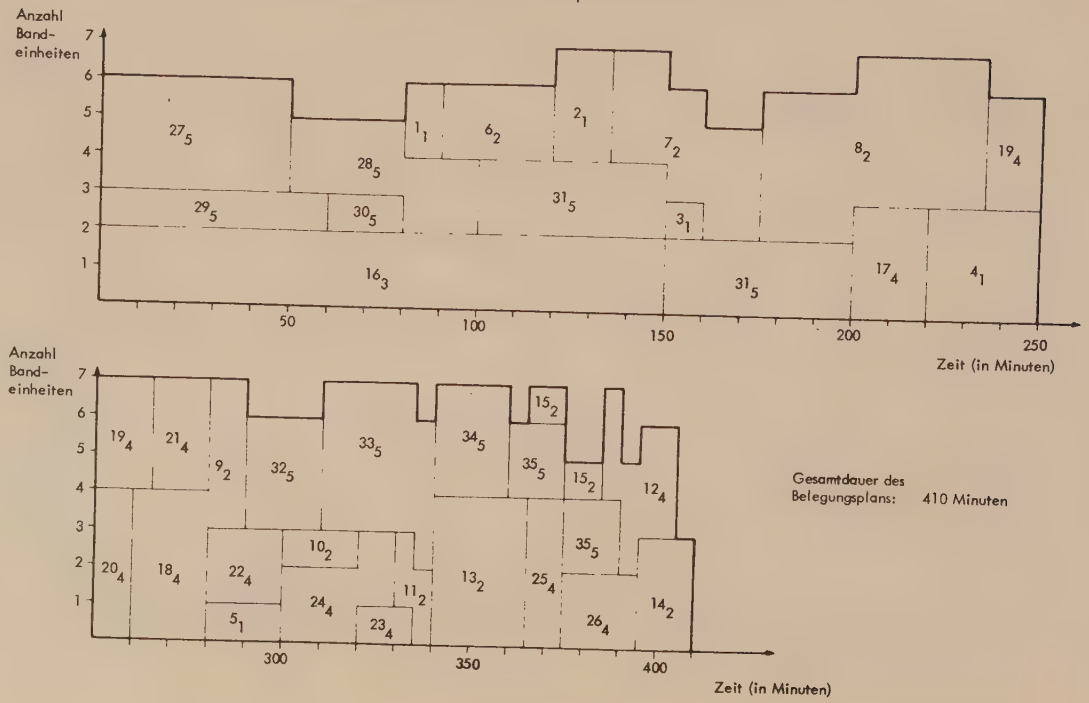
(1) Blocknummer
(2) Rang des Jobs
(3) Anzahl der Folgejobs innerhalb des Blocks
(4) Kurzbezeichnung des Jobs
(5) Auftraggeber des Jobs
(6) Programmierer-Nr.
(7) Sachbearbeitergruppe
(8) Jobbezeichnung
(9) Anzahl der zu ladenden Dateien
(10) Verweilzeit des Jobs
(11) Druckausgabeklassen
(12) Druckzeit für die Druckausgabeklassen
(13) benötigte Regiongröße
(14) Bereitstellungstermin
(15) Jobklasse (Anzahl Bandeinheiten)
(16) Anzahl der Druckbänder
(17) Druckzeit der Druckbänder
(18) benötigter Plattenspeicherplatz in Zylindern
(19) Abgabetermin
(20) Angabe der Folgejobs
(21) Angabe der Jobs, die gleiche Eingabedatenbestände verwenden

Anhang 3 : Ergebnis der Zeitrechnung in Minuten für das Beispiel in Abb. 32

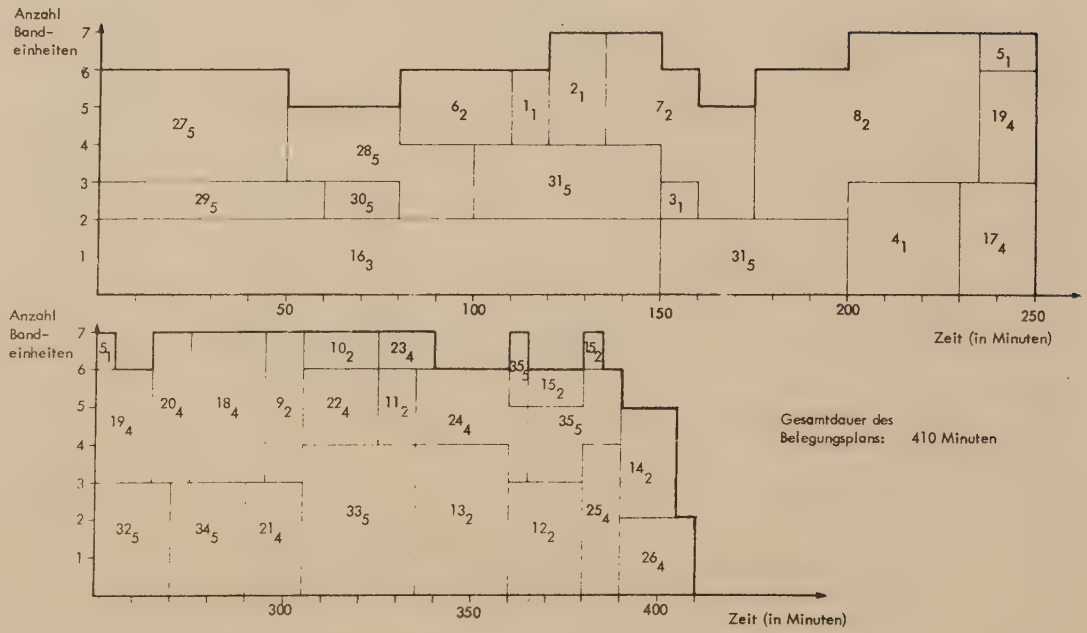
Jobname	Verweilzeit	Frühester Anfangs-Zeitpunkt FAZ	Frühester End-Zeitpunkt FEZ	Spätester Anfangs-Zeitpunkt SAZ	Spätester End-Zeitpunkt SEZ	Gesamte Pufferzeit GP
1	10	0	10	195	205	195
2	15	10	25	205	220	195
3	10	25	35	220	230	195
4	30	35	65	230	260	195
5	20	65	85	260	280	195
6	30	0	30	135	165	135
7	40	0	40	155	195	155
8	60	30	90	165	225	135
9	10	40	50	195	205	155
10	20	50	70	205	225	155
11	10	90	100	225	235	135
12	20	100	120	260	280	160
13	25	100	125	235	260	135
14	15	100	115	265	280	165
15	20	125	145	260	280	135
16	150	0	150	130	280	130
17	20	0	20	170	190	170
18	20	20	40	190	210	170
19	30	0	30	180	210	180
20	10	0	10	185	195	185
21	15	10	25	195	210	185
22	20	40	60	210	230	170
23	15	60	75	235	250	175
24	30	60	90	230	260	170
25	10	75	85	250	260	175
26	20	90	110	260	280	170
27	50	0	50	0	50	0
28	50	50	100	50	100	0
29	60	0	60	20	80	20
30	20	60	80	80	100	20
31	100	100	200	100	200	0
32	20	200	220	200	220	0
33	30	220	250	220	250	0
34	20	220	240	230	250	10
35	30	250	280	250	280	0

Die hohen Pufferzeiten der Blöcke 1 - 4 sind darauf zurückzuführen, daß die 5 Blöcke als Gesamtnetz verarbeitet wurden. Dadurch gehen die spätesten Termine der Blöcke 1 - 4 von der Gesamtdauer des zeitlich längsten Blockes (5) aus.

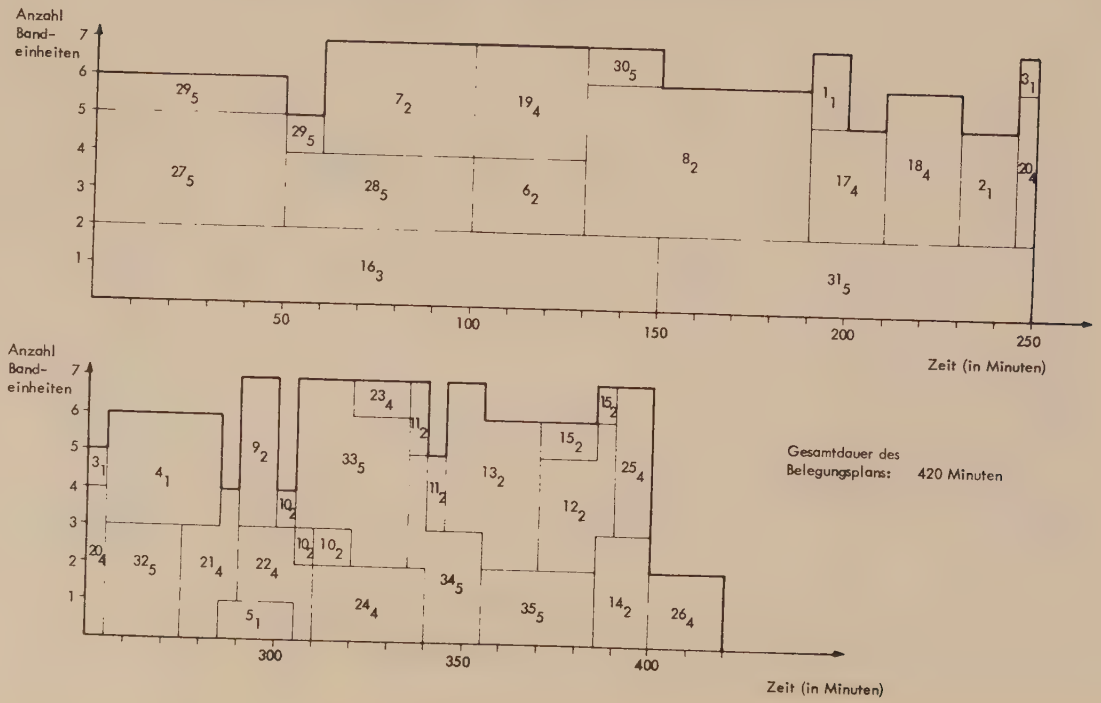
Anhang 4 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedrige Pufferzeit, 2. niedrige Verweilzeit, und 3. niedriger spätester Endzeitpunkt



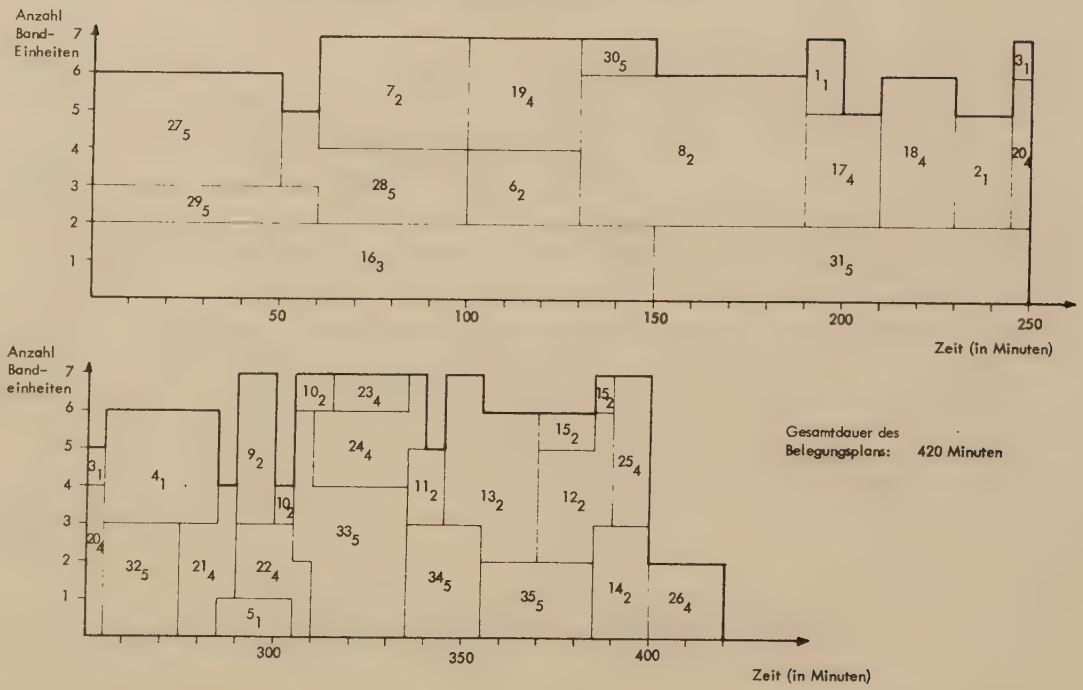
Anhang 5 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedrige Pufferzeit, 2. hohe Verweilzeit und 3. niedriger frühester Anfangszeitpunkt



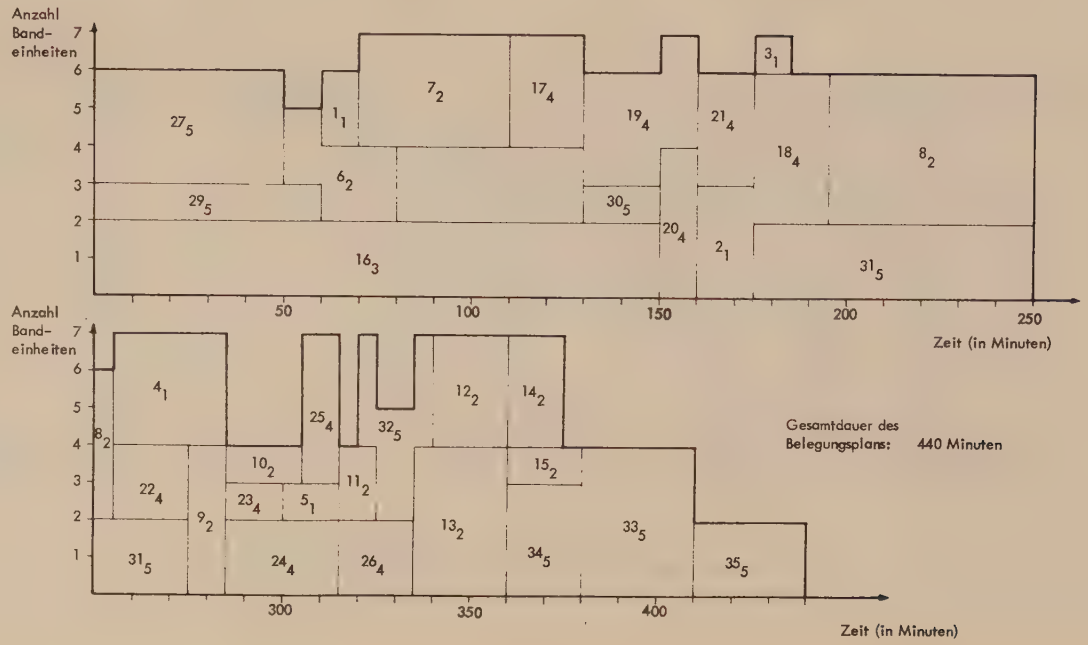
Anhang 6 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. hohe Blockdauer, 2. hohe Verweilzeit und 3. niedrige Pufferzeit



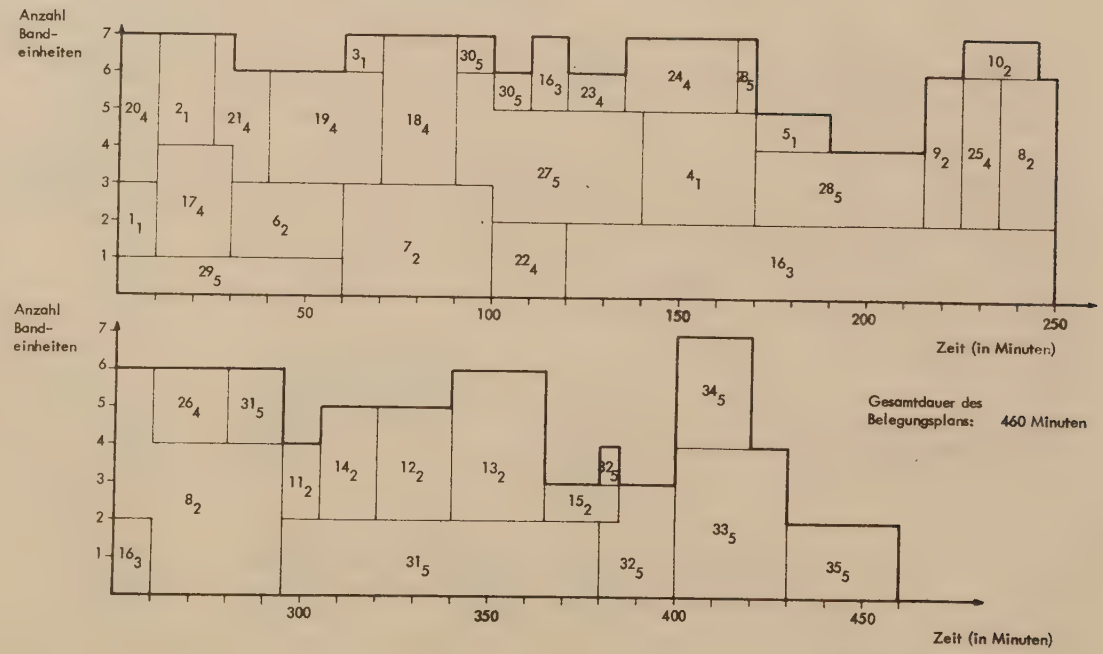
Anhang 7 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. hohe Verweilzeit, 2. niedrige Pufferzeit und 3. niedriger spätester Endzeitpunkt



Anhang 8 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedriger frühester Anfangszeitpunkt, 2. niedrige Pufferzeit und 3. hohe Verweilzeit



Anhang 9 : Einplanungsergebnis für das Beispiel aus Abb. 32 bei der Prioritätsregelkombination 1. niedriger frühester Endzeitpunkt, 2. niedrige Pufferzeit und 3. hohe Verweilzeit



VERZEICHNIS DER ABKÜRZUNGEN

CPU	Central Processing Unit (Zentraleinheit)
DAB 3	Programm zur Dynamischen Anlagenbelegung
DA-Datei	Datei im Direkt-Zugriff
DCB	Data Control Block
DD-Statement	Datei-Definitions-Steuerkarte
EDV	Elektronische Datenverarbeitung
EDVA	Elektronische Datenverarbeitungsanlage
E-/A-	Ein-/Ausgabe-
EXEC Statement	Steuerkarte zur Definition eines Job-Steps
EXEC 8	Executive Betriebssystem von UNIVAC
FAZ (FAT)	Frühester Anfangs-Zeitpunkt (-Termin)
FEZ (FET)	Frühester End-Zeitpunkt (-Termin)
GP	Gesamtpuffer (gesamte Pufferzeit)
Job Statement	Job Definitions Steuerkarte im OS
KB	Kilo-Byte (1024 Bytes)
OS	Operating System der IBM
OS-MFT	Multiprogramming with a Fixed Number of Tasks
OS-MVT	Multiprogramming with a Variable Number of Tasks
PBS	Platte-Betriebssystem von SIEMENS
Region	Kernspeicherbereich
RUN Statement	Job Definitions Steuerkarte des EXEC 8
SAZ (SAT)	Spätester Anfangs-Zeitpunkt (-Termin)
SEZ (SET)	Spätester End-Zeitpunkt (-Termin)
sog.	sogenannte
TCB	Task Control Block
u.a.	unter anderem
UPRO	Unterprogramm

VERZEICHNIS DER ABKÜRZUNGEN ZU DEN ABLAUFDIAGRAMMEN IN KAPITEL 6

A'	vorläufige minimal erreichbare Anschaltzeit
A^*	minimal erreichbare Anschaltzeit
B_{ax} (B_{ax*})	Bandeinheitenanforderung des Jobs x (x*)
B_b	belegte Bandeinheiten zu einem bestimmten Zeitpunkt
B_v	verfügbare Bandeinheiten zu einem bestimmten Zeitpunkt
BF_1, BF_2, BF_3	Belegungsfaktoren für Kernspeicher-Regions, Magnetbandeinheiten und Magnetplattenspeichereinheiten
BF_{max}	maximaler Belegungsfaktor
BF_m	mittlerer Belegungsfaktor
BF_{min}	minimaler Belegungsfaktor
D_B	Differenz zwischen belegten und verfügbaren Bandeinheiten
D_K	Differenz zwischen belegten und verfügbaren Kernspeicher-Regions
D_P	Differenz zwischen belegten und verfügbaren Plattenspeicherbereichen
E	Anzahl der Bandeinheiten, um die erhöht werden darf
E_B	Maximalerhöhung der Anzahl der Bandeinheiten ($B_v - B_b$)
E_K	Maximalerhöhung der Anzahl der Kernspeicher-Regions ($K_v - K_b$)
E_P	Maximalerhöhung der Anzahl der Plattenspeicherbereiche ($P_v - P_b$)
i_t	Anzahl der Jobs, die zu einem bestimmten Zeitpunkt eingeplant werden können
K_{ax} (K_{ay})	angeforderte Kernspeicher-Region des Jobs x (y)
K_b	belegte Kernspeicher-Regions zu einem bestimmten Zeitpunkt
K_v	verfügbare Kernspeicher-Regions zu einem bestimmten Zeitpunkt
P_{ax} (P_{ay})	angeforderte Plattenspeicherbereiche des Jobs x (y)
P_b	belegte Plattenspeicherbereiche zu einem bestimmten Zeitpunkt
P_v	verfügbare Plattenspeicherbereiche zu einem bestimmten Zeitpunkt
R_x ($R_{x'}$)	Restzeit des Jobs x (x')

LEBENSLAUF

Hans-Jürgen Kottmann	4600 Dortmund-Scharnhorst, Spannstr. 24
Geboren am	21.1.1945 in Weidenau/Sieg
Staatsangehörigkeit	deutsch
Eltern	Ernst Kottmann, Facharzt für Hals-, Nasen- und Ohrenkrankheiten und Ehefrau Anneliese, geb. Rittershausen
Schulen	April 1951 bis April 1955 Volksschulen in Weidenau und Dortmund; April 1955 bis Februar 1964 Leibniz-Gymnasium in Dortmund, mathematisch-naturwissenschaftlicher Zweig
Reifeprüfung	21.2.1964
Studiengang	Studium der Volkswirtschaftslehre vom SS 1964 bis einschließlich SS 1965 an der Universität Münster, WS 1965/66 an der Universität München, SS 1966 bis zum SS 1969 an der Universität Münster
Studiumabschluß	Prüfung für Diplom-Volkswirte am 11.4.1969
Berufspraxis	seit dem 1.6.1969 System-Organisator in der Zentralen Datenverarbeitung der Hoesch AG in Dortmund